THE ORIGINAL COMPILER

オリジナル・コンパイラ

新言語作成の技法

大貫広幸 著

新言語作成の技法

まえがき

プログラム言語の作成技法について書かれた本は何冊か発行されています．ただ，その多くはミニコンより上位のコンピュータを対象にしているようです．そのため，コンパイラを高水準言語で記述していたり，かなりレベルが高かったりでよく理解できないといったこともあるようです．

たしかに本格的なコンパイラを作成するためには，多くの知識とプログラミング・テクニック，そして経験といったものが必要です．しかし，パソコンで文字どおりパーソナルなコンパイラをつくるためには，それほどの大変な力量が必要かというと，決してそうではありません．

本書では，Z80CPUをターゲットに，コンパイラを作成するための基本的な手法をまとめてみました．ですから,Z80のアセンブラ言語やPascalといった言語をある程度,理解されている方なら十分読みこなせる内容になっています．

これからコンパイラをつくろうと思っている方やプログラム言語の構造を研究しようとしている方に本書が少しでもお役に立てば幸いです．

本書第2部ではコンパイラの作成例として全ソース・プログラムを掲載しています．これはディスクでも読者サービスをいたしますのでご利用ください(詳細は巻末を)．

また，第2部3章のPC-8801バージョンのコンパイラおよびライブラリは大熊英男，石塚圭樹両氏によって移植・作成されたものです．ご協力に対し，ここに感謝を述べておきます．

■本書の構成■

本書は，１部と２部の２部構成になっています。第１部ではコンパイラを作成するための基本的な手法を，第２部では作成例として全ソース・リストを掲載します。各章の要約は次のとおりです。

●第１部

第１章　コンパイラとは

コンパイラそのものの構造と付随するリンカやＯＳなどのソフトウェアの概要を示します。

第２章　言語の設計

言語の表現方法として，読みやすく理解しやすい『構文図』の書きかたを中心に解説します。

第３章　コンパイラの設計と手法

本書の中核となる章です。コンパイラを作成するための基本的な手法をまとめています。PascalやFORTRANを例にあげ，どうオブジェクト・プログラム（Ｚ80のアセンブラ言語で記述）を展開するかを中心に解説します。また，構文解析や式のコンパイルは身近な言語であるBASICで記述した例も示します。引数の処理など部分的に少々難しい部分もありますが，なるべく図で示して理解しやすいように配慮しました。

●第2部

第１章　プログラム言語Stellar

コンパイラの作成例として『Stellar』と名付けたコンパイラを示します。本章は文法書を書くときの参考にもしてください。

第２章　CP/MバージョンのStellarコンパイラ

CP/M上で動作するStellarコンパイラの全ソース・リストを掲載します。研究の対象としてプログラムの解読は多くの情報をもたらすでしょう。

第３章　PC-8801バージョンのStellarコンパイラ

PC-8801に移植したStellarコンパイラと数多くのライブラリ，サンプル・プログラムを示します。

CONTENTS

第1部 ─ コンパイラの基本的作成技法 ─

第1章　コンパイラとは ― 3

1-1　プログラム言語 ― 4
1-1-1　プログラム言語の分類 ― 4
1-1-2　インタープリタとコンパイラ ― 6
1-2　コンパイラの処理過程と構造 ― 8
1-2-1　コンパイルから実行まで ― 8
1-2-2　コンパイラの構造 ― 10
1-3　リンカ ― 16
1-4　ランタイム・ルーチン ― 22
1-5　OSとコンパイラ ― 24

第2章　言語の設計 ― 27

2-1　オリジナル言語をつくるために ― 28
2-2　言語の表しかた ― 32
2-3　文法書をつくる ― 37

第3章　コンパイラの設計と手法――39

3-1　コンパイラの仕様をまとめる――40
3-2　字句解析――45
3-2-1　ソース・プログラムの読み込み――45
3-2-2　字句の解析――46
3-3　式のコンパイル――58
3-3-1　式の構文解析の方法――58
3-3-2　逆ポーランド記法を用いて式をコンパイルする――59
3-3-3　再帰的な方法で式をコンパイルする――80
3-3-4　代入文と単項演算子,配列,関数――96
3-4　記号表と宣言文――109
3-4-1　記号表の構成――109
3-4-2　記号表の検索と登録――110
3-4-3　パスと表管理――116
3-4-4　宣言文の処理とラベル――118
3-4-5　全域的な名前と局所的な名前の処理――122

3-5　制御文のコンパイル——127
3-5-1　条件文——127
3-5-2　繰り返し文——137
3-5-3　複合文——149
3-5-4　GOTO文——150
3-5-5　サブルーチン(手続き)の呼び出し——150
3-6　オブジェクト・プログラムのメモリへの割りつけ——152
3-6-1　ロケーション・カウンタ——152
3-6-2　宣言文とメモリの割りつけ——155
3-7　サブルーチンと関数の処理——160
3-7-1　引数の渡しかたと引数の処理——160
3-7-2　サブルーチンと関数のコンパイル——174
3-8　コンパイラ全体の構成——180
3-9　コンパイラの作成手順——183

第2部 ―コンパイラの作成例―

第1章 プログラム言語Stellar―189

1-1 プログラム言語Stellarとは―190
1-2 Stellarの設計思想―191
1-3 Stellarの構文と文法―192
1-4 エラーメッセージ―231

第2章 CP/MバージョンのStellarコンパイラ―235

2-1 コンパイラの使用法―236
2-2 サンプル・プログラム―242
2-3 全プログラム・リスト―249

第3章 PC-8801バージョンのStellarコンパイラ―373

3-1 コンパイラの使用法―374
3-2 ソース・プログラムのつくりかた―377
3-3 使用上の注意―378
3-4 ライブラリの使いかた―380
3-5 サンプル・プログラム―399
3-6 コンパイラのダンプ・リストと打ち込みかた―418

第 1 部

第1章

コンパイラとは

　コンパイラとはいったい何をするものか，コンパイラをとりまく実行環境との関連を含めて，コンパイラについての基本的な事項を解説します。

1-1 プログラム言語

プログラム言語と一口にいっても種々雑多の言語があります。そこで，まず最初に言語を分類し，次に言語の処理方式であるインタープリタ方式とコンパイラ方式の違いについて述べます。

1-1-1 プログラム言語の分類

プログラム言語 (programming language) は，人間がコンピュータを動かすためにつくり出した言語で，人工言語 (artificial language) とも呼ばれています。これに対し，日本語や英語のことを自然言語 (natural language) といいます。

プログラム言語を使用目的，分野別に分類すると，一般的に**図1－1**のようになります。図からわかるように大きく汎用言語 (general purpose language) と特殊用言語 (special purpose language) に分けられます。汎用言語は，名前が示すとおり広範囲な分野で使われる言語ですが，それぞれの言語にはやはり得手不得手があります。この他に，高水準言語と低水準言語という分けかたもあります。この分けかたは，あくまで相対的なもので，より自然言語に近いプログラム言語を高水準言語，より機械（コンピュータ）に近い言語を低水準言語と呼んでいます。

低水準言語は**図1－1**の機械向き言語のことを指しますが，アセンブラを低水準言語と呼ぶ人はあまりいないようです。

図1-1 プログラム言語の分類

分類	説明
◆プログラム言語	（コンピュータを動かすための人口言語のことを言う　）
●汎用言語	**（広範囲な分野で使用できる言語　）**
機械向き言語	（コンピュータ固有の命令が使われる　）
機械語	（命令を２進数，16進数で表す　）
アセンブラ言語	（命令を人間にもわかるように記号に表した言語　）
	（数式や英文に近い形でプログラムできる　）
手続き型言語	（BASIC，FORTRAN，COBOL などの言語がある　）
非手続き型言語	（RPG などの言語がある　）
●特殊用語	**（特定の分野で使用される言語　）**
数値制御用言語	（工作機械のためのプログラム言語，代表的な言語にＡＰＴ（アプト）がある）
ＣＡＤ（キャド）言語	（回路，機械，自動車などの設計に用いられる言語。たとえば電子回路設計用にはＥＣＡＰ（イーキャップ）がある。）
ＣＡＩ（シーエーアイ）言語	（教育システム用の言語）
プロセス制御用言語	（大きいものでは化学，石油などのコンビナート，小さいものでは温度などの制御を行う言語）
シミュレーション言語	（アナログ・コンピュータで行われていた微分方程式を用いた連続系をデジタル・コンピュータで行うための言語，CSMP などがある）
リスト処理言語	（リストと呼ばれるデータ構造を扱う言語，リスト処理は非数値計算(記号処理)に向いている。LISP が有名である）
ストリング処理用言語	（文を調べたり，文を変換するようなストリング処理を行うための言語，SNOBOL（スノーボール）などがある）
数式処理用言語	（数学的な数式を扱うために作られた言語，微積分，多項式などが計算できる MACSYMA という言語がある）
コンパイラ記述用言語	（コンパイラを記述するための言語，C 言語のもとになった BCPL が有名である）
⋮	

1-1-2　インタープリタとコンパイラ

高水準言語は，実行時の処理方法によりインタープリタとコンパイラに分けられます。パソコンでは，一般にBASICインタープリタが使われています。インタープリタはプログラムを解釈しながら直接実行し，コンパイラはプログラムを一旦機械語に変換してから実行します。

たとえば，

```
K＝2＊(L／5—N)
```

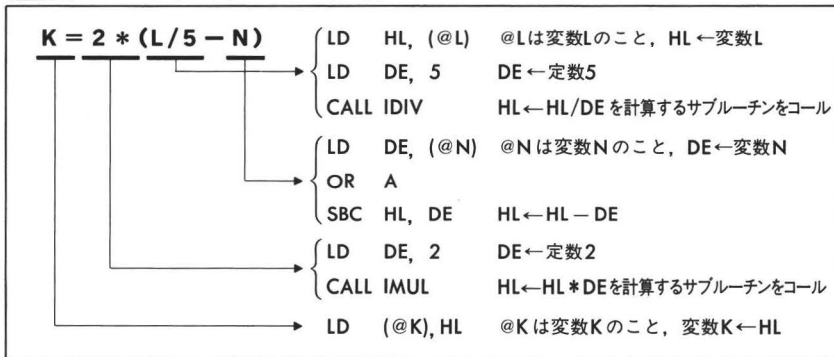
という式があるとすると，インタープリタでは次のように解釈・実行されます。

①“K＝”により，この文は代入文であり，計算結果を格納する変数はKであると解釈。
②“2＊（”よりカッコ内の式を計算し，2倍すると解釈。
③“L／5”は変数Lを5で割ると解釈し，変数表を検索してLの値を求め，計算する。
④“－N”は変数表よりNの値を求め，③の計算結果から減算する。
⑤“)”より②で解釈した内容を実行。
⑥計算結果を変数表の変数Kの値とする。

一方，コンパイラは文を字句ごとに解析し，機械語に変換します。**図1－2**は同じ式をコンパイルした例です。ただし，変数K，L，Nを2バイトの整数とし，機械語ではなくZ80のアセンブラ言語で記述しています。

インタープリタとコンパイラを比較すると，次のようなことが言えます。

①実行時のプログラムは，コンパイラでつくった方が数倍～数10倍高速です。これはインタープリタが文の解釈

図1-2 K＝2＊(L/5－N)のコンパイル例

と実行を同時に行っているのに対し，コンパイラは文の解釈に相当する部分を一括して行い，実行時には解釈が必要ないので，この分だけ実行時間が速いのです。
②インタープリタの方がプログラム開発の効率が良いようです。プログラム開発時は変更や修正が頻繁に行われるので，すぐに実行できるインタープリタの方が向いています。コンパイラの方は，実行するまでの操作が多いため，すぐに結果が見られません。

以上のことから，開発はインタープリタで行い，完成したらコンパイラでコンパイルして実行するときは機械語で行うようにすれば開発効率も高く，完成したプログラムの実行時間も速いということになります。しかし，現状では（特にパソコンでは）こういうシステムはほとんど見かけません。一部BASICのように両方備わっているものもありますが，コンパチビリティがなかったり，BASIC言語そのものがかかえる問題点などで必ずしも十分とは言えないようです。

1-2 コンパイラの処理過程と構造

ここでは，コンパイラがどのようにして高水準言語を機械語に変換するのか，その処理過程と構造について解説します。

1-2-1　コンパイルから実行まで

高水準言語で書かれたソース・プログラムをコンパイルして得られるプログラムをオブジェクト・プログラムと呼びます。オブジェクト・プログラムの形式には主に次の種類があります。

①すぐに実行できる機械語
②再配置情報＋機械語とした再配置可能（リロケータブル）なプログラム
③アセンブラのソース・プログラム

CP/M や MS-DOS などの DOS 上で走行するコンパイラやミニコン以上のコンピュータでは，ほとんどが②か③の形式です。

①はディスクなどの高速な外部記憶装置がない場合に使われます。また，この形式のコンパイラはプログラムのサイズも小さく，コンパイラの基本動作を知る上でも手頃な教材なので，本書第 2 部で記載したコンパイラもこの形式です。

図 1－3 はコンパイルから実行までの過程を図で示したものです。ローダはディスクなどに出力したプログラムをメモリにロードし実行するためのプログラムです。リンケージ・エディタ（これはリンカとも呼ばれ，パソコンではこの呼びかたが多いのでこれ以降，本書ではリンカとします）やリンケージ・ローダは，再配置情報か

図1-3 コンパイルから実行までの過程

① 直接機械語を出力するコンパイラ（ファイルをあまり使わないのでOSは必ずしも必要なし）

ソース・プログラム
高水準言語のプログラム（ファイル）
入力
コンパイル
コンパイラ
出力
オブジェクト・プログラム
機械語のプログラム（ファイル）
入力
ローダ
実　行

オブジェクト・プログラムをメモリ上に直接出力した場合はローダは必要ない。

② 再配置可能なオブジェクト・プログラムを出力するコンパイラ（ファイルを多く使うのでOSが必要となる）

高水準言語のプログラム（ファイル）
入力
コンパイラ
出力
オブジェクト・プログラム
再配可能なオブジェクト・プログラム（ファイル）
リンケージ・ローダ
実　行
入力
リンケージ・エディタ（リンカ）
出力
機械語のプログラム（ファイル）
入力
ローダ
実　行

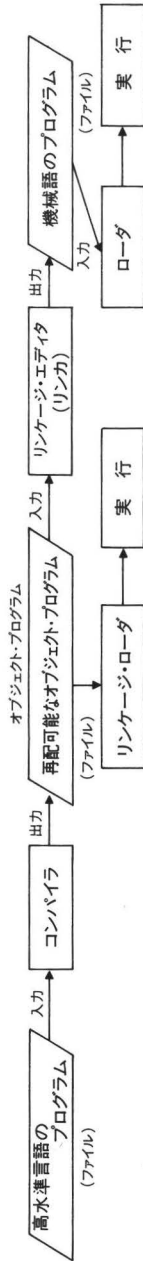
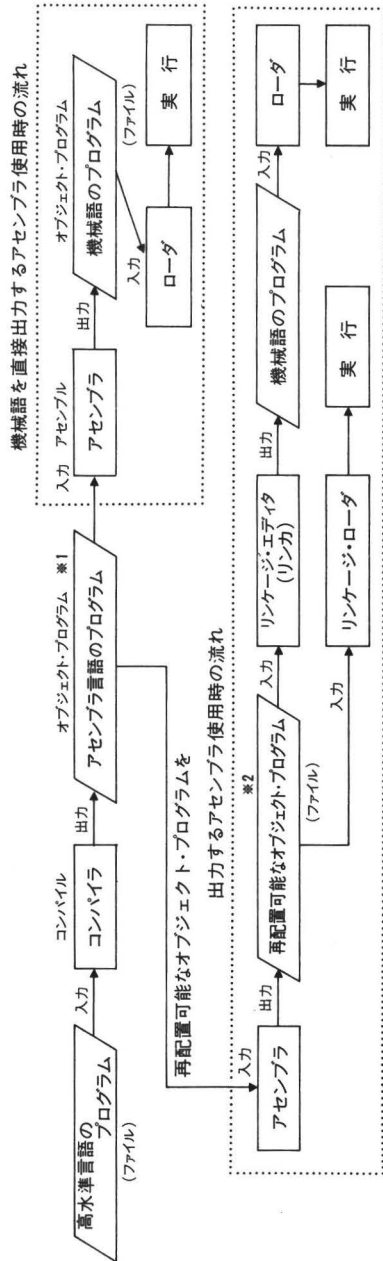
③ アセンブラ言語のプログラムを出力するコンパイラ（ファイルを多く使うのでOSが必要となる）

高水準言語のプログラム（ファイル）
入力
コンパイル
コンパイラ
出力
オブジェクト・プログラム　※1
アセンブラ言語のプログラム

機械語を直接出力するアセンブラ使用時の流れ

入力
アセンブル
アセンブラ
出力
オブジェクト・プログラム
機械語のプログラム（ファイル）
入力
ローダ
実　行

再配置可能なオブジェクト・プログラムを出力するアセンブラ使用時の流れ

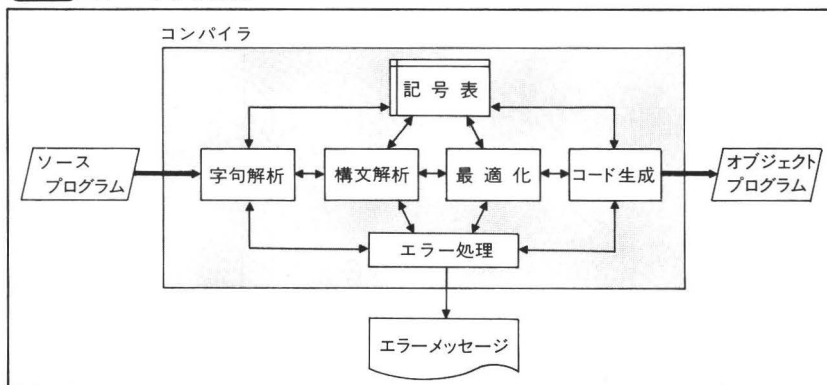
入力
アセンブラ
出力
※2
再配置可能なオブジェクト・プログラム（ファイル）
入力
リンケージ・エディタ（リンカ）
出力
機械語のプログラム
入力
ローダ
実　行
入力
リンケージ・ローダ
実　行

注）※1　コンパイラから見れば，オブジェクト・プログラムであるがアセンブラから見ればソース・プログラムである。
※2　アセンブラが出力した，オブジェクト・プログラム

ら実行可能な機械語プログラムをつくるものです。リンカと再配置情報については，“**1 - 3 リンカ**”のところで解説します。リンケージ・ローダはリンカとローダを一緒にしたようなものです。

再配置可能なオブジェクト・プログラムを出力するプログラムの特徴は，1つのソース・プログラムを何本かに分けてコンパイルして，最後にリンカによって1本の機械語プログラムにできることです。このため大きなプログラムを数人でつくる場合などに有効です。また，オブジェクト・プログラムの形式が同一であれば，複数の言語を使っての開発もできます。たとえば，高水準言語だけではどうしても記述できない部分や処理を高速にしたいルーチンなどをアセンブラ言語で記述するといったこともできます。

1-2-2 コンパイラの構造

それでは，コンパイラはどのようにしてソース・プログラムからオブジェクト・プログラムをつくっているのでしょうか。一般にコンパイラの中では図1-4のような処理でソース・プログラムからオブジェクト・プログラムがつくられています。これらの処理は互いに独立して

図1-4 コンパイラの構造

いるのではなく，密接に関係しあっています。

では，コンパイラ内の各処理が何を行っているか各処理別に見ていくことにしましょう。

①字句解析（lexical analysis）

ソース・プログラムの文字列を読み，トークン(token)と呼ばれる語句に分解する処理が字句解析です。トークンは予約語や名前（変数名やサブルーチン名，関数名，ラベルなど），定数（数値や文字）などのことで，次の構文解析部で使います。

たとえば，BASIC コンパイラは

```
250 IF A>B THEN 200
```

というソース・プログラムを字句解析部で

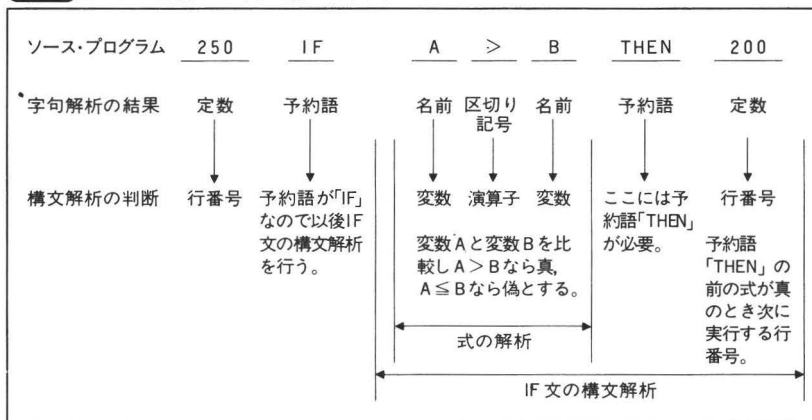
定数（行番号）“250”
予約語　“IF”“THEN”
名前（変数名）“A”“B”
区切り記号（演算子）“>”
定数　“200”

というように分解します。

②構文解析（parsing, syntax analysis）

構文解析は，字句解析によって得られたトークンによってステートメント（命令）の構成を調べ，後の最適化処理やコード生成処理へ解析結果を渡す処理です。構文解析ではまず字句からステートメントの種類を判断した後，さらに詳細なステートメント別の構文解析を行います。**図1－5** は BASIC 言語の IF 文がどのように構文解析されるかを示した例です。

字句解析と構文解析はプログラム言語の文法書に示されている構文にしたがって解析していきます。この段階で文法にしたがわない構文は誤り（エラー）としてはじ

図1-5 BASIC言語のIF文の構文解析例

かれ，エラー処理でメッセージが出力されます。

構文解析，特に式の解析には，逆ポーランド記法を使う方法や再帰的に解析していく方法などがあります。

③最適化（optimization）

最適化は，構文解析の結果から最適な(ムダの少ない)オブジェクト・プログラムをつくる処理で，次のコード生成処理では最適化によって選ばれた機械語をオブジェクト・コードとします。最適化の主な目的は，オブジェクト・プログラムのサイズを小さくし，実行速度を上げることです。同じ言語のコンパイラでも，この最適化のしかたによってオブジェクト・プログラムのサイズや実行速度が異なり，ひいてはコンパイラ自体の性能(評価)を決める要因ともいえます。

たとえば，“A＋B＊2－1”という式をコンパイルし，8086CPUの命令をオブジェクト・コードとして出力する場合，式のとおりにコンパイルすると，

```
MOV  AX, @A          ;レジスタAXへ変数Aをロ
                      ード，@Aが変数Aを示す。
```

```
PUSH AX          ;スタックへレジスタAXの
                  内容をプッシュする。
MOV  AX, @B      ;レジスタAXへ変数Bをロ
                  ード，@Bが変数Bを示す。
MOV  BX,2        ;レジスタBXへ定数2を設
                  定。
MUL  BX          ;レジスタAXとレジスタ
                  BXを乗算し，結果はレジス
                  タAXへ入れる。オーバー
                  フローは無視。
MOV  BX, AX      ;“B＊2”の結果をレジス
                  タBXへ転送。
POP  AX          ;スタックより変数Aの内容
                  をポップする。
ADD  AX, BX      ;レジスタＡＸにレジスタ
                  BXの内容を加算する。
MOV  BX, 1       ;レジスタＢＸへ定数1を設
                  定。
SUB  AX, BX      ;レジスタＡＸよりレジスタ
                  BXの内容を引く。
```

となります。簡単な最適化の例として，元の式を“Ｂ＊２＋Ａ－１”と変形し，さらに“＊２”を左へのシフト命令，“－１”をデクリメント命令にすれば

```
MOV  AX, @B      ;レジスタAXへ変数Bをロ
                  ード。
SAL  AX, 1       ;レジスタAXを左へシフト
                  する。これはAX＊2と同
                  じ。
ADD  AX, @A      ;レジスタAXへ変数Aの内
                  容を直接加算する。
```

```
DEC  AX              ;レジスタAXをデクリメン
                      ト（AX－1）。
```

とオブジェクト・コードが小さくなり，実行する命令数も減るので実行速度も速くなります。

④コード生成（code generation）

コード生成は，構文解析や最適化の結果からオブジェクト・コードをつくる処理です。ここでつくられるオブジェクト・コードは，そのコンパイラによって，機械語命令であったり，ある特定のアセンブラの形式を備えたニモニック列であったりします。機械語プログラムを直接出力するのか，アセンブラ形式のニモニックを出力するのか，あるいは再配置可能なオブジェクト・プログラムを出力するのかによって，コンパイル後の処理が“**1－2－1　コンパイルから実行まで**”の説明のように変ってきます。

オブジェクト・コードはそれ自体一つあるいは複数の命令であり，オブジェクト・コードが集まってオブジェクト・プログラムとなります。

⑤エラー処理（error handling）

構文解析で検出された構文上の誤りなどは，このエラー処理でエラーメッセージとして出力し，知らせます。その後エラー処理ではエラーの種類を調べ，重大なエラーだったとき以外はエラーに適当な処理を施し，コンパイルを続行します。この一連の処理をエラー回復(error recovery）といいます。

エラー回復の処理は非常に難しく，どのようにしてエラーを処理しコンパイルを続行するかが問題です。たとえば，エラーがある文は終わりまで読みとばし，次の文から改めてコンパイルを続行する方法，エラーがある文の前後の文脈からエラーの原因を推測してコンパイルを

続行する方法などがあります。この場合，前者の方はわりと簡単にできますが，後者の方はかなり難しくエラー回復の処理だけでもかなり大きくなってしまいます。いずれにせよ，エラー回復を正しく行わないとコンパイルを再開したとき正しい構文なのにエラーと判断されてしまうことが起きたりします。

⑥記号表（symbol table）

記号表は変数やサブルーチン，関数，ラベルなどの名前とアドレス，属性などを記憶する表です。コンパイル中はこの記号表への登録，参照が繰り返されるため，記号表の操作の速度がコンパイル全体の速度に大きな影響を及ぼします。とりわけ表の参照は登録よりも頻繁に起きるため，記号表の検索を速くすることがコンパイル速度の向上につながります。

1-3 リンカ

コンパイラやアセンブラがオブジェクト・プログラムとして再配置可能なプログラムを出力する場合，リンカが必要になります。リンカの仕事は**図1-6**に示すように複数個の再配置可能なオブジェクト・プログラムをリンク（link＝連結）し，1本の実行可能な機械語プログラムをつくることです。

リンカはモジュール（module）と呼ばれる単位でリンクします。再配置可能なオブジェクト・プログラムは一つ以上のモジュールの集まりで，モジュールは一つ以上のルーチンの集まりです。リンカはリンクのときにライブラリ（library）より必要なモジュールを取り出してリンクします。この場合のライブラリは再配置可能なオブジェクト・プログラムのモジュールを集めたものです。

図1-7は**図1-6**をモジュール単位で見た場合の例です。図では，オブジェクト・プログラムA，B，C（モジュールA，BX，BY，C）をすべてリンクし，ライブラリからサブルーチンβ2を含むモジュールβがリンクされることを示しています。この場合，必要のないサブルーチンβ1，β3もリンクされています。

それでは，今度は再配置可能なオブジェクト・プログラムの中身とリンクはどのように行われるか見ていきましょう。

コンパイルするとき，アドレスはハードウェア上の絶対アドレスではなく，セグメント（segment）という論理的な単位でプログラムを分割し，セグメントの先頭をゼロとする相対アドレスを使います。コンパイルされるプログラムは大きく分けて，変数などのデータの部分と式や制御文などの命令の部分に分けられます。コンパイラ

図1-6 リンカの動作

図1-7 リンカによるモジュールのリンク

オブジェクト・プログラムA

モジュールA

メイン・ルーチン

サブルーチンA1

サブルーチンA2

モジュールAではサブルーチンβ2が必要

オブジェクト・プログラムB

モジュールBX

サブルーチンB1

サブルーチンB2

モジュールBY

サブルーチンB3

オブジェクト・プログラムC

モジュールC

サブルーチンC1

リンカ

ライブラリ

モジュールα

サブルーチンα1

サブルーチンα2

サブルーチンα3

サブルーチンα4

モジュールβ

サブルーチンβ1

サブルーチンβ2

サブルーチンβ3

・・・

機械語プログラム

メイン・ルーチン

サブルーチンA1

サブルーチンA2

サブルーチンB1

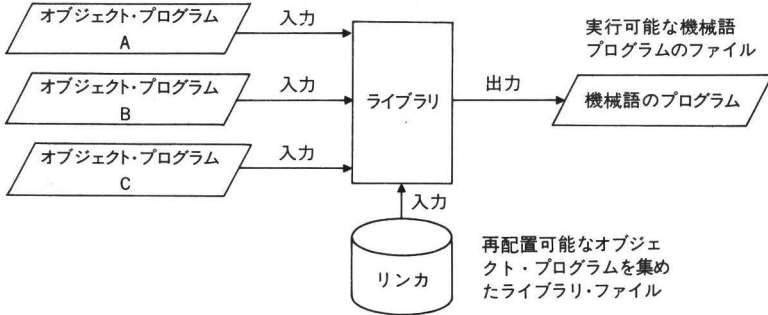
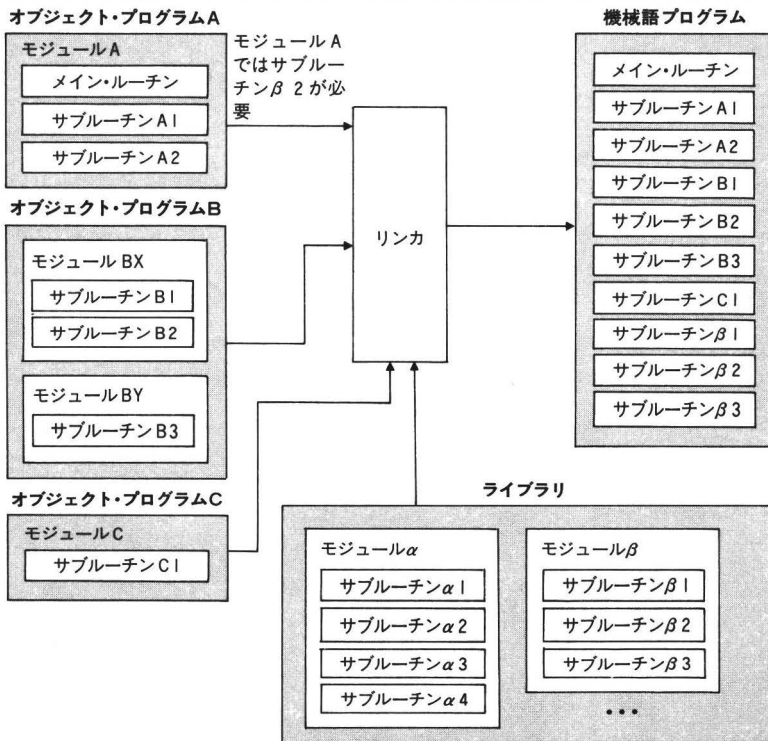
サブルーチンB2

サブルーチンB3

サブルーチンC1

サブルーチンβ1

サブルーチンβ2

サブルーチンβ3

は，この二つの部分にそれぞれセグメントを設け，コード生成のときデータの部分のオブジェクト・コードをデータ・セグメントと呼ばれるセグメントへ出力し，命令の部分はコード・セグメントと呼ばれるセグメントへ出力します（**図1-8**）。

さらに，コード生成部ではオブジェクト・コードをつくるときに，それらが何であるかを示すコードを付けるようにしています。**表1-1**はその一例です。

この表を例にとると，リンカは次のような処理をしていきます。

●TYPE 0 00XX

XXの1バイトを直接機械語の1バイトとする。

●TYPE 1 01XXXX

XXXXの2バイトがコード・セグメントの相対アドレスなので，“現コード・セグメントの先頭の絶対アドレス＋XXXX”で求めた2バイトを機械語2バイトとする。

●TYPE 2 02XXXX

XXXXの2バイトがデータ.セグメントの相対アドレスなので，“現データ・セグメントの先頭の絶対アドレ

図1-8 再配置可能なオブジェクト・プログラムの構造

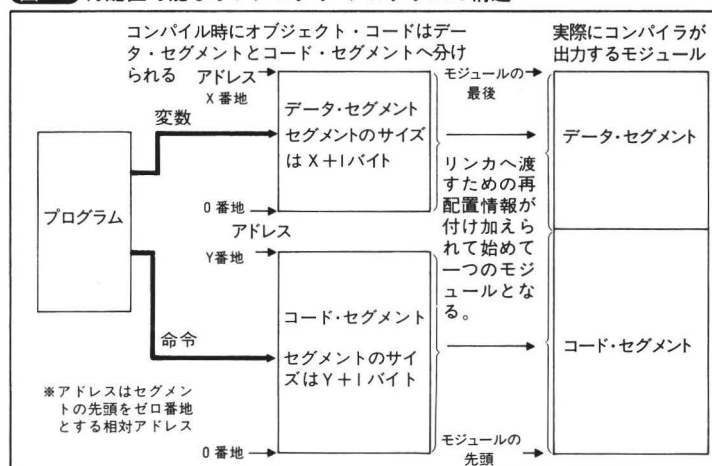
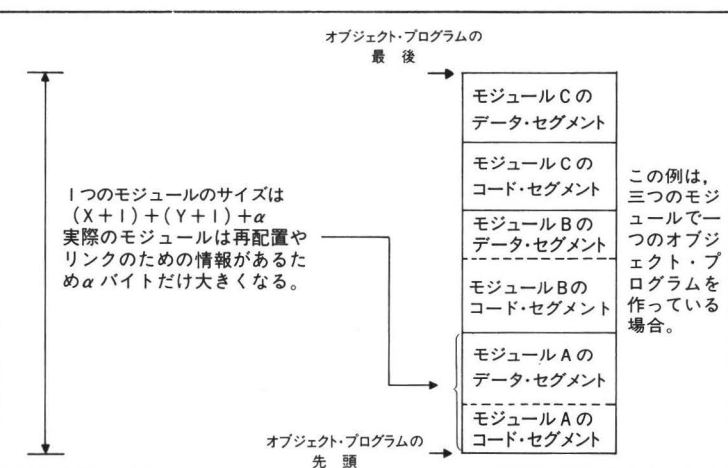



表1-1 再配置可能なオブジェクト・コードの例

オブジェクト・コード	内容
●TYPE 0 [0 0][X X] (XX: 1バイトの値、00: オブジェクト・タイプのコードを示す)	機械語の命令や定数などの1バイトの値。
●TYPE 1 [0 1][X X][X X] (XXXX: 2バイトの相対アドレス)	コード・セグメントの相対アドレスを示す2バイトの値。
●TYPE 2 [0 2][X X][X X] (XXXX: 2バイトの相対アドレス)	データ・セグメントの相対アドレスを示す2バイトの値。
●TYPE 3 [0 3][Y Y][Y Y]・・・[Y Y][0 0] (YY…YY: 外部記号名)	他のモジュールにある外部記号名の値を示す。リンクにより2バイトの値となる。
●TYPE 4 [0 4][X X][X X][Y Y]・・・[Y Y][0 n] (XXXX: 2バイトの値、YY…YY: 外部記号名)	・n＝0 の場合 2バイトの値を名部記号名で定義する。 ・n＝1 の場合 2バイトの値がコード・セグメントの相対アドレスとして外部記号名で定義される。 ・n＝2 の場合 2バイトの値がデータ・セグメントの相対アドレスとして外部記号名で定義される。
●TYPE 5 [0 5]	コード・セグメントの始まりを示す。
●TYPE 6 [0 6]	データ・セグメントの始まりを示す。
●TYPE 7 [0 7][Y Y][Y Y]・・・[Y Y][0 0] (YY…YY: ライブラリ名)	ライブラリを指定する。
●TYPE 8 [0 8]	モジュールの終わりを示す。

※[ ]内の2桁の数字や英字は，2桁の16進数で表わされる1バイト。
※X X はアドレスや値などを示す。
※Y Y はASCIIなどの文字コードを示す。

ス＋XXXX”で求めた2バイトを機械語2バイトとする。

●TYPE 3 03YYYY…YY00

YY…YYは1バイト以上の文字列で外部記号名を表しているので，リンカは等しい外部記号名がすでに定義されていれば，その外部記号名が示す2バイトを機械語2バイトとする。定義されていなければ定義されるまで処理を保留し，定義されたときに処理の続きを行う。

●TYPE 4 04XXXXYYYY……YY0n

n＝0の場合，XXXXの2バイトの値をYY…YYの1バイト以上の文字列で表される外部記号名で定義する。

n＝1の場合，XXXXは2バイトのコード・セグメントの相対アドレスなので，“現コード・セグメントの先頭の絶対アドレス＋XXXX”で求めた2バイトの値をYY…YYの外部記号名で定義する。

n＝2の場合，XXXXは2バイトのデータ・セグメントの相対アドレスなので，“現データ・セグメントの先頭の絶対アドレス＋XXXX”で求めた2バイトの値をYY…YYの外部記号名で定義する。

●TYPE 5 05

現時点の絶対アドレスをコード・セグメントの先頭アドレスとして記憶しておく。この値はTYPE1と4のオブジェクト・コードのところで計算に使われる値である。

●TYPE 6 06

現時点の絶対アドレスをデータ・セグメントの先頭アドレスとして記憶しておく。この値はTYPE2と4のオブジェクト・コードのところで計算に使われる値である。

●TYPE 7 07YYYY……YY00

YY…YYはライブラリ名でリンカを終了するときに未定義の外部記号名があった場合，指定されたライブラリを自動的に検索し，未定義の外部記号名が定義されているモジュールをリンクする。

●TYPE 8 08

現在リンクしているモジュールのリンクを終了し，つぎのリンクへ移る。

以上，リンカと再配置可能なオブジェクト・プログラムの例を見てきましたが，これはあくまで仮想的なものです。実際に使われているものは，もっと複雑です。パソコンでは，米マイクロソフト社の8ビットCPU(80系の8080，8085，Z80)用のアセンブラやコンパイラで使われているオブジェクト・コードの形式が有名です。これはマニュアルに書かれているので，機会があれば一度読んでみてください。

1-4 ランタイム・ルーチン

コンパイラが出力した機械語のオブジェクト・プログラムは実行するときにランタイム・ルーチン (run time routine)と呼ばれるものが必要です。ランタイム・ルーチンは実行時に必要なサブルーチンやデータの集まりで，ディスク，キーボード，CRTなどの入出力や演算，関数，文字列の操作といったサブルーチンがあります。ちなみに，ランタイムとはプログラムを実行している時間のことです。

コンパイラが再配置可能なオブジェクト・プログラムを出力する場合，リンカはランタイム・ライブラリから実行に必要なサブルーチンのみ抜粋してリンクします（図1－9）。リンカのところで述べたように，モジュール単位でリンクされるため，一つのモジュールに多くのサブルーチンを入れると最終的にプログラムも大きくなってしまいます。

コンパイラが直接機械語のオブジェクト・プログラムを出力する場合，ランタイム・ルーチンは，

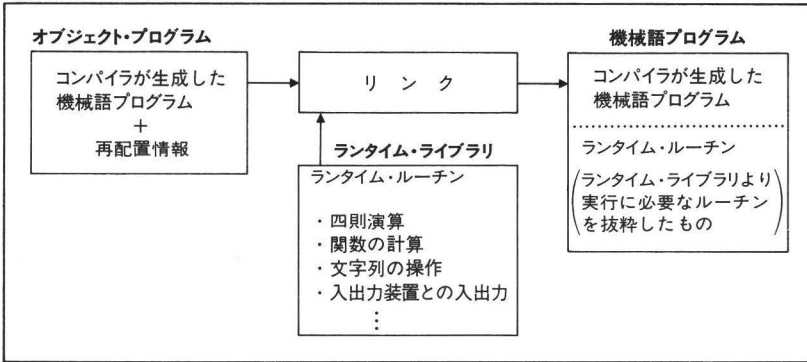
①コンパイル時にオブジェクト・プログラムの一部としてランタイム・ルーチンも出力する。

②ランタイム・ルーチンを常にメモリの一定の領域に記憶しておく。

③オブジェクト・プログラムの中に外部記憶装置からランタイム・ルーチンをロードするプログラムを入れておき，はじめにこのルーチンでランタイム・ルーチンをメモリにロードする。

④③とは逆にランタイム・ルーチンで外部記憶装置よりオブジェクト・プログラムを読み込む。

などの方法で機械語プログラムからランタイム・ルーチンを使えるようにします。

図1-9 ランタイム・ルーチンとのリンク

機械語を直接出力するコンパイラは，コンパイルするときにハードウェア上の絶対アドレスを使います。したがって，ランタイム・ルーチン内の各サブルーチンの呼び出しアドレスも絶対アドレスで決められています。

ランタイム・ルーチンは数命令の機械語で処理できないものをサブルーチン化したものです。たとえば整数の四則演算は短い機械語でできるが，実数の四則演算には長いルーチンが必要であるといった場合，コンパイル時に整数演算のときは直接機械語を，実数演算のときはランタイム・ルーチンを呼び出す機械語をつくるようにします。これによってオブジェクト・プログラムのサイズを小さくすることができます。

1-5 OSとコンパイラ

OS とはオペレーティング・システム（operating system）のことで，コンピュータを効率的に使いやすく運営，操作するために用意されたソフトウェア体系のことです。**図1-10**に示すようにアセンブラやコンパイラといったプログラムもOSの一部なのです。狭い意味では制御プログラム（管理プログラムともいう）のことをOSという場合もあります。制御プログラムは，ほかにモニタ（monitor），スーパバイザ（supervisor）とかシステム・プログラム（system program）とも呼ばれています。

図1-10 OSの体系

- OS
  - 制御プログラム
    - ジョブ管理
    - タスク管理
    - データ管理
  - 処理プログラム
    - 言語処理プログラム
      - FORTRANコンパイラ
      - COBOLコンパイラ
      - アセンブラ
      - ・
      - ・
    - サービス・プログラム
      - ユーティリティ・プログラム
      - リンカ
      - デバッガ
      - ・
      - ・
    - ユーザー作成プログラム
  - アプリケーション・プログラム
    - データ・ベース
    - ワード・プロセッサ
    - 作表計算パッケージ

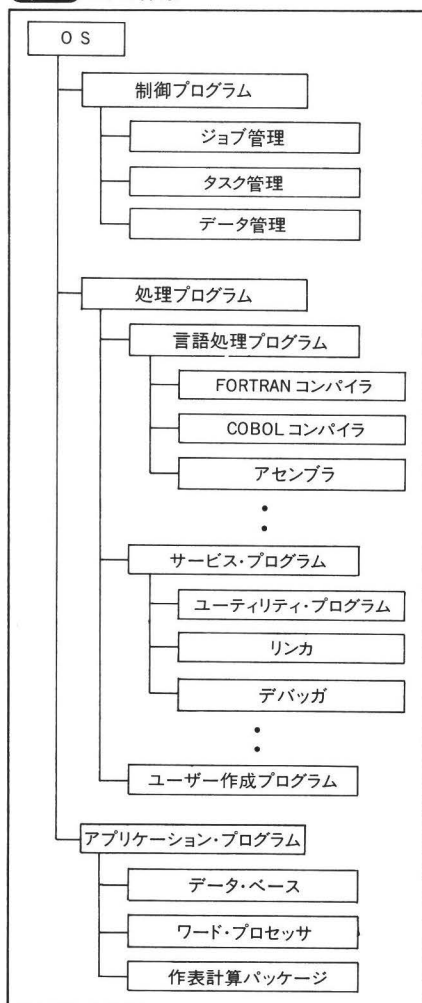
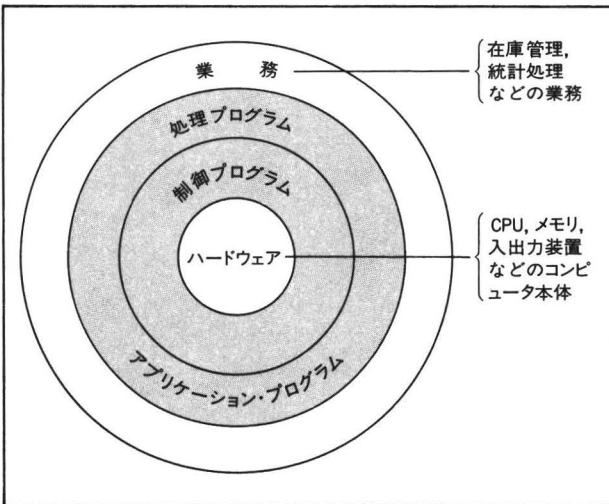
OSの概念図は**図1-11**のようになっています。この図は，内側の機能は外側によって利用されることを意味しています。つまり，ハードウェアは制御プログラムから利用され，制御プログラムは処理プログラムなどから利用されることを示します。このことは，処理プログラムなどからの制御プログラムの利用方法さえ同じなら，ハードウェアが異なっても実行できるということです。つまり，OSが同じであれば，他のコンピュータの処理プログラムも利用できることになります。

OS上で走るコンパイラをつくる場合，入出力やタスク管理の制御を行うためにはシステム・コールまたはスーパバイザ・コールと呼ばれるOSによって定められた一定の手続きを踏む必要があります。特に，マルチタスクの機能をもつOSでは，この規則を守らないと正常に動作しないばかりか暴走することさえあります。

図1-11 OSの概念

第2章

言語の設計

プログラム言語を設計するうえで，何をどう設計するのか，また設計時に念頭においてほしいことを述べていきます。

2-1 オリジナル言語をつくるために

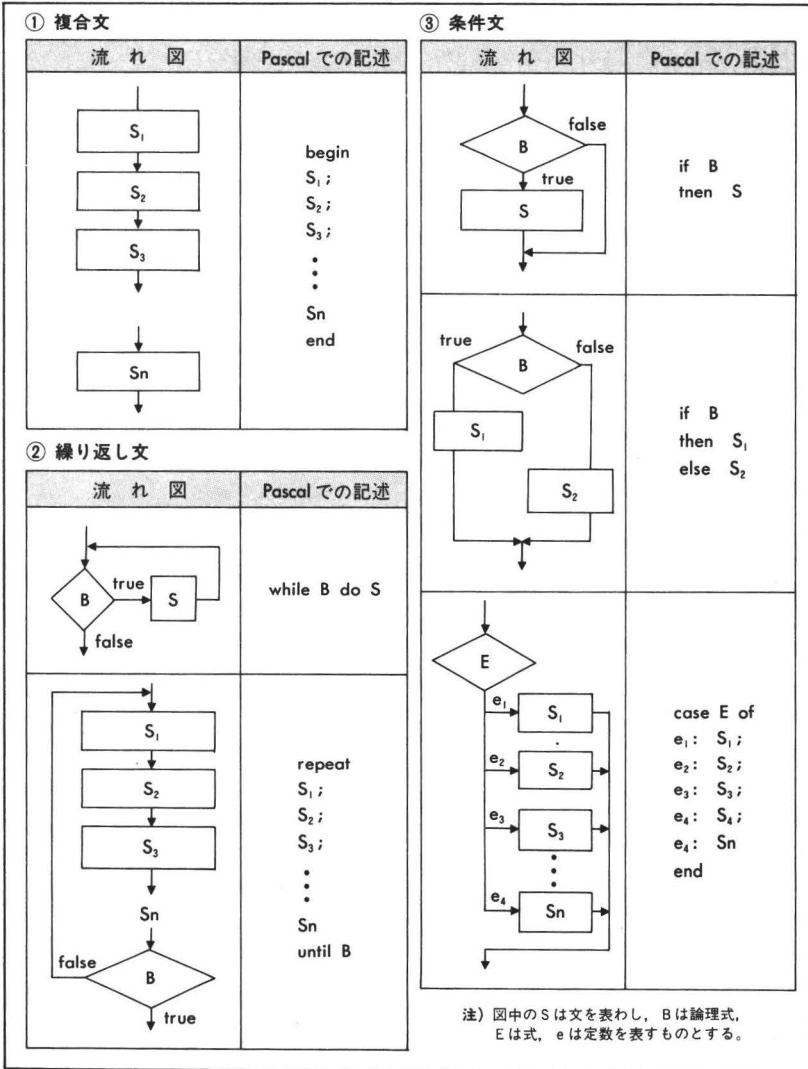
“オリジナル”といっても，既存の言語とまったく異なるものをつくることは，個人のレベルではかなり難しいでしょう。また，既存の言語のコンパイラもほとんどがミニコン以上のコンピュータを対象に文法がつくられているために，これもパソコンで実現するには，かなりの労力を要します。個人のレベルでは，やはり，既存の言語に大幅に制約を加えたものか，似たような文を持つ言語を新たにつくる程度になるでしょう。

いずれにしろプログラム言語は，コンパイラやオブジェクト・プログラムが実行される環境を考えて設計します。また，文はなるべく自然言語に近づけ，一目で一応意味がわかる程度にまとめます。もちろん，どういう目的でコンパイラをつくるのか，1機種専用にするか移植を考えてつくるか，前もって決めておきます。

新しく言語を設計するときに，考慮してほしいことを列挙してみます。

①構造化プログラミングのための文をつくる。

構造化プログラミングは，ソフトウェアの質や信頼性を上げるための手法で，プログラムの作成やデバッグが楽になります。構造化プログラミングは図2－1に示すような形式でプログラムをつくることで，これによって不必要なGOTO文がなくなります。さらに，段階的詳細化（トップ・ダウン）ができるというのも大きな特徴です。段階的詳細化というのは，大まかなことから始めてだんだん詳細のレベルに持っていくことで，プログラムの大きな変更が少なくなり，信頼性の高いソフトウェアをつくることができます。

図2-1 構造化プログラミング

②サブルーチンや関数に独立性を持たせる。

BASIC言語では，変数に独立性がなく，おなじ変数名であれば他のルーチンで変更されてしまいます。これでは長いプログラムや数人でプログラムをつくるときに大

変不便です。

サブルーチンや関数は，始めと終わりをはっきりさせて，特定のサブルーチン内でのみ使える変数（これをローカル変数とか局所的変数という）とメイン・ルーチンとサブルーチンで共通に使える変数（これをグローバル変数とか全域的変数という）の2種類の変数を持つようにします。こうすると他のルーチンの変数を誤って変えてしまうという過ちが起きにくくなります。

③再帰的プログラミングがつくれるようにする。

これは必ずしも必要ないと思いますが，サブルーチンや関数を再帰的 (recursive) な呼び出しができると便利な場合があります。再帰的な呼び出しとは，サブルーチンや関数が実行されているときに，直接あるいは間接的に自分自身を呼び出して実行することです。

再帰的プログラムは，人間の思考過程と同じようなことをさせるプログラムをつくる場合に有効です。そのた

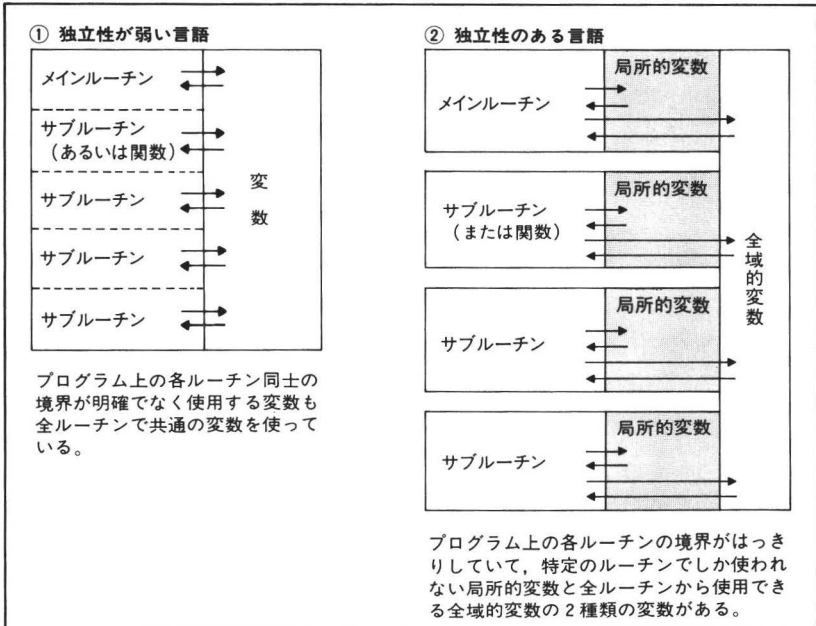
図2-2 サブルーチンや関数の独立性

① 独立性が弱い言語

メインルーチン
サブルーチン（あるいは関数）
サブルーチン
サブルーチン
サブルーチン
変数

プログラム上の各ルーチン同士の境界が明確でなく使用する変数も全ルーチンで共通の変数を使っている。

② 独立性のある言語

プログラム上の各ルーチンの境界がはっきりしていて，特定のルーチンでしか使われない局所的変数と全ルーチンから使用できる全域的変数の2種類の変数がある。

め，人工知能の分野ではよく使われる手法です。

④機械語やハードウェアを直接操作できる文を用意する。

パソコンでは，メモリやI/Oを直接操作したい場合がどうしてもでてきます。そのため，メモリやI/Oへの直接書き込み・読み出しを行う文，機械語を呼び出すための文，直接機械語を書くための文などがあると便利です。

このほか，数Kバイト程度の小さなコンパイラをつくるときは，

①扱えるデータ（変数や演算）の型を1種類に限定する。この場合，実数より整数のほうがより小さくできる。

②配列は1次元のみとする。

③変数名やサブルーチン名といった名前の有効文字数を短くする。

④記号表に登録できる名前の数を制限する。

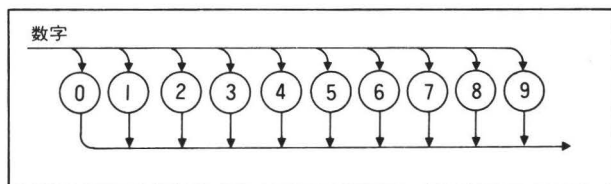
⑤Pascalなどの言語のようなブロック構造（サブルーチンや関数の中にそこだけで使われるサブルーチンや関数が書ける構造）にしない。

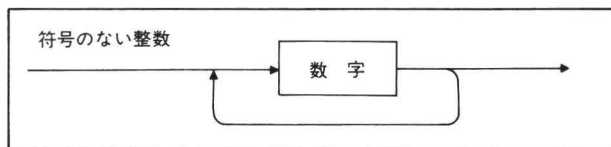
といったことも考えられます。

2-2 言語の表しかた

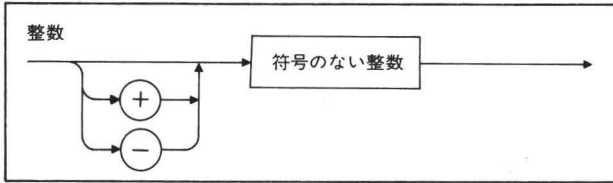
プログラム言語を設計する場合，言語の構文を明確に表さなければなりません。構文の表し方にはバッカス記法（Backus Naur form：BNF と略される）や構文図（syntax diagram）などが使われます。バッカス記法は少しわかりづらく，最近はあまり使われません。これに対し，構文図は読みやすく書きやすいという特徴があり，さらにフローチャートなどにすぐに変換できるため広く使われています。ここでも構文図について述べることにします。

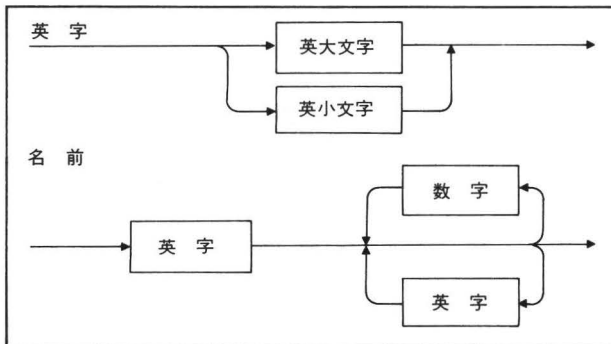
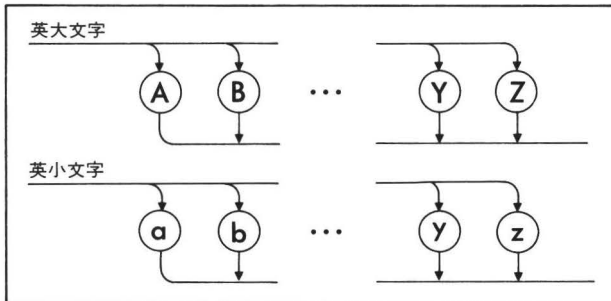
プログラム言語の構文を直観的に理解しやすいように表す記述法が構文図です。

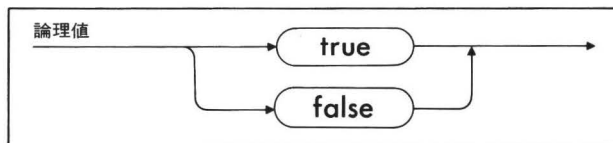
たとえば，構文図では数字を次のように表現します。

これは，数字とは，0または1，または2，…または9であるという意味です。この円の中には1文字の文字や記号を書きます。これを終端記号と呼びます。矢印のとおり進んで得られた結果が，図の左上に描いたものの意味となります。構文図の入口と出口は必ず1個ずつです。

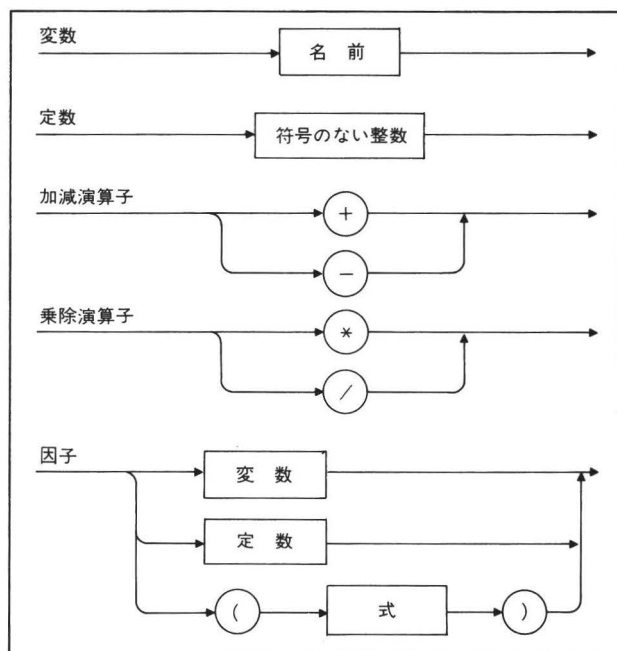
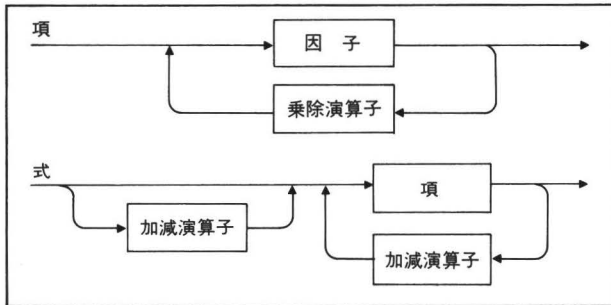
符号のない整数は次のようになります。

ここで，長方形は中のものが別のところで表現してあることを意味しています。これを非終端記号と呼びます。

また，矢印がループになっていますが，これは符号のない整数とは，数字がいくつか並んだものであるということを意味します。

次に，整数は，

となります。これは整数とは，符号のない整数，またはそのまえに“＋”か“－”のいずれかを付けたものであるという意味です。

変数名や関数名といったプログラム中で使われる名前を「英字で始まり，その後に英字か数字が続いたもの」とすると次のようになります。

英大文字，英小文字のように，それが何であるか明らかなときは省略してもかまいません。

次の例は論理値の構文図です。

これは，論理値とはtrueまたはfalseという文字列で表すことを意味します。また，楕円は，円と同じ意味に使われます。

最後の例として四則演算とカッコのみを使う式の構文を表してみます。演算の優先順位はカッコのなかが一番高く，次に乗除算，そして加減算の順とします。定数は符号のない整数とします。

ここで“A＋２＊B”という文字列が式として正しい構文になっているかこの図から調べてみましょう。

式の構文図をみると“ A”は加減演算子ではないので，項であることがわかります。項の構文図を見ると因子となっているので，因子の構文図を見ます。因子は変数，定数，カッコのどれかですが，変数，名前それぞれの構文図を見ると結局“A”という名前の変数であることがわかります。さらに構文図を逆に戻ると“A”は因子だとわかります。さらに項へ戻って，“A”は因子で，次の文字の“＋”は項の構文図の中の乗除演算子ではないので，“A”までが項となります。

次の“＋”は，式の構文図から加減演算子だとわかります。この矢印をさらにたどると，“２”以下の文字列は項でなければなりません。項，因子，乗除演算子の構文図を見ると，“２”は因子，“＊”は乗除演算子，そして“B”は因子だとわかります。これで“２＊B”が項であるとわかります。

式の構文図へ戻り，全体を整理すると“A”は項であり，“２＊B”が項だとわかったので，結局，項と項とを加減演算子で結んだ“A＋２＊B”は式であることがわかります。

では，“A＋＊B”は式といえるでしょうか。これは構文図で調べると“A”は項，“＋”は加減演算子とわかりますが，次の“＊B”が項といえません。項の最初は因子のはずですが，“＊”は因子ではありません。

このように，構文図をたどれば，文字列が文法上正しいかどうかわかります。

また，“A＋2＊B”という式では“A”と“2＊B”が項で，“＋”が加減演算子ですから，計算の順序は，“(A＋2）＊B”でなく“A＋（2＊B)”であることもわかります。

文章で説明したので，少々複雑ですが，構文図と見比べればわかると思います。

2-3 文法書をつくる

プログラム言語は構文図を使って表しますが，これだけで言語のすべてが表せるわけではありません。たとえば，構文図のところで例としてあげた名前や符号のない整数は，このままでは長い名前でも何桁の整数でもよいことになってしまいます。実際には，名前は16文字まで，などといったようにいろいろな制限がつきます。このように構文図だけでは表せないことも，文法書にはまとめておきます。

文法書には，“プログラムの構成”，“サブルーチンや関数の宣言”，“文”，“標準関数”……というように項目別にまとめます。また，文法書にはあいまいなところを残さず，できる限りこまかいところまで書いておくと，後の作業が楽になります。

コンパイラを制作中に文法書は何度か修正が加わるのが普通なので，完成したときに文法書を書きなおすことになります。このとき，他人が読んでプログラムをつくれる程度に図や例を入れて書かなくてはなりません。しかし，これは大変な仕事なので，個人レベルでは“できる限り”ということになります。

第3章

コンパイラの設計と手法

コンパイラをつくるときに必要な式や制御文のコンパイル方法，記号表の管理といった基本的な手法について解説します。

3-1 コンパイラの仕様をまとめる

コンパイラをつくる前には仕様書をつくります。仕様書には

①コンパイラの方式
②コンパイラの実行環境
③ソース・プログラムの形式
④オブジェクト・プログラムの形式
⑤コンパイラの操作方法
⑥コンパイラに付随するソフトウェアの仕様
⑦出力するエラーメッセージとエラー回復の方法
⑧その他，必要と思われること

といったことを書いておきます。

この仕様書をつくるときは，対象となるプログラム言語の文法だけでなく，実行環境をよく考えなければなりません。同じプログラム言語であっても，仕様によって操作方法やオブジェクト・プログラムの形式，生成されるオブジェクト・コードの大きさ，実行速度などが異なってきます。

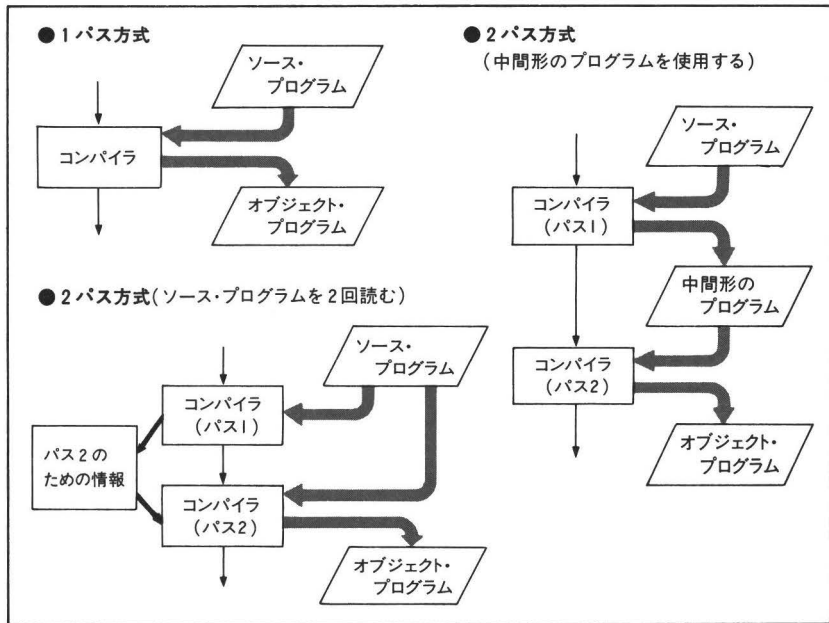
それでは，仕様書の中に書かれる個々の内容について見ていきましょう。

①コンパイラの方式

1－2－1で述べたとおり，コンパイラが出力するオブジェクト・プログラムの形式によって，コンパイル後の処理過程が異なります。ですから，まずコンパイラがどのようなオブジェクト・プログラムを出力するかを決めなければなりません。

次に，パス (pass) の回数を決めます。パスとはソース・プログラムからオブジェクト・プログラムをつくるまでに通る処理の各段階のことです。1回のパスでソー

ス・プログラムが1回読み込まれます。ソース・プログラムを1回読み込んだだけでオブジェクト・プログラムをつくる方式を1パス方式，2回読み込む方式を2パス方式といいます。中には3パス以上のものもあります。各パスをパス1，パス2といった具合に呼びます。2パス以上の方式のコンパイラの中には，ソース・プログラムを読み込むのはパス1だけで，パス2以降は前のパスで出力した中間形のプログラムだけを読み込む方式のものもあります。

図3-1 1パス方式と2パス方式

②コンパイラの実行環境

コンパイラの実行環境を明確にします。OSを使うならその種類とバージョン，パソコンの機種名，最低必要なメモリ容量（できればメモリ・マップも）と外部記憶装置，一時的に使うファイルとその編成などは明らかにせねばなりません。また，コンパイラ以外に必要とする

プログラムは何か(たとえばエディタやリンカなど)，またそれらのうちで新たにつくらなければならないものはどれか，といったソフトウェア環境もまとめておきます。

③ソース・プログラムの形式

ここではコンピュータの中（メモリ上やファイル上）でのソース・プログラムの形式をはっきりさせます。

OS下のコンパイラなら，ソース・プログラムはそのOSに付属のエディタで入力され，テキスト・ファイルとして記憶されます。そして行の区切りやファイルの終わりの印には，各OSが独自に定義している制御文字が使われます。そのため，OS下で走るコンパイラでは，ソース・プログラムの形式をそのOS標準のテキスト・ファイルの形式に合わせなければなりません。CP/MなどのOSの場合，文字コードはJISかASCIIコード，テキスト・ファイルの終わりを示す文字にはSUB(16進で1A)が使われ，行の区切りにはCR，LF(16進で0D，0A)の2文字が使われています。

OSを使わない場合には，ソース・プログラムをBASICインタープリタのエディタを使ってつくるか，新たに専用のエディタを作成するかのいずれかになります。BASICインタープリタのエディタを使うなら，コンパイラ側はそのエディタがソース・プログラムを記憶する形式（一般的にポインタというものを含み多少複雑）に合わせるしかありません。専用のエディタを作成するならソース・プログラムの形式を自由に決めることができるので2文字以上の空白を圧縮したりといったことができます。

④オブジェクト・プログラムの形式

コンパイラが出力するオブジェクト・プログラムの形式（メモリまたはファイル上での形式）を決めます。

たとえばアセンブラ言語のプログラムを出力するコン

パイラなら，オブジェクト・プログラムはそのアセンブラのソース・プログラムの形式にしなければなりません。再配置可能なプログラムを出力するコンパイラなら，前出の**表1-1**(19ページ) のように機械語や再配置情報，リンク情報がどのようになっているかまとめておけばよいでしょう。直接機械語を出力するコンパイラは，実行時に必要になる特殊コードや値をオブジェクト・プログラム中に埋め込んで出力する場合があります。これらがオブジェクト・プログラム上にどのように配置され，何を意味し，どう表現されているのかをまとめておかなければなりません。

⑤コンパイラの操作方法

ここでは，コンパイラの起動方法，コンパイラに対するいろいろな指定(オプションとかスイッチと呼ばれる)のしかたと意味など，起動から終了までの操作方法をまとめておきます。

⑥コンパイラに付随するソフトウェアの仕様

コンパイラ以外に作成しなければならないプログラムがあればその仕様も決めます。たとえばコンパイラが再配置可能なプログラムを出力するならリンカが必要になります。OS 標準または市販されているものを使わずに専用のリンカを作成するなら，そのリンカの仕様書もつくらねばなりません。

⑦エラーメッセージと，エラー回復の方法

コンパイラが出力するエラーメッセージと意味，そしてエラーに対するエラー回復の処理をまとめておきます。この段階でエラーメッセージをまとめておくとコンパイルのエラーチェックの設計が楽になります。

⑧その他，必要と思われるすべてのこと

これは，コンパイラの設計に必要と思われること，たとえば将来の拡張予定や他のコンピュータへの移植のための設計上の注意といったことなどです。

以上のような事柄をまとめ，一冊の仕様書をつくります。この仕様書をもとにしてコンパイラの概要設計，詳細設計をし，コーディング，デバッグそして完成となります。作業を進めていくうちに仕様書の内容と食い違ってきたら，仕様書を修正するように心がけてください。

3-2 字句解析

これより，コンパイラの設計に必要ないろいろな手法を，BASIC や FORTRAN，Pascal といった言語を例にとりながら解説していきます。ただし，解説のため正式の規格や文法から多少逸脱しているところがあるので注意してください。また，解説の中で使っている機械語(アセンブラ言語）はZ80用(8ビット CPU)です。

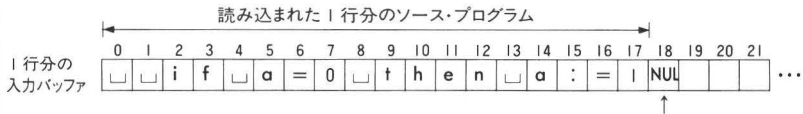
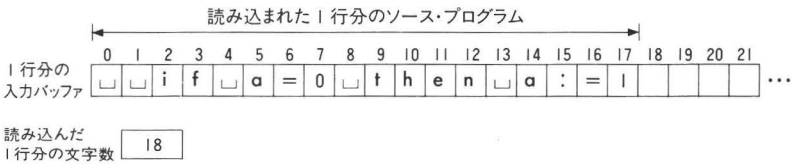
3-2-1 ソース・プログラムの読み込み

ソース・プログラムはパソコンの場合，ファイルかメモリに記憶されているのが一般的です。ファイル上にソース・プログラムがある場合は**図3-2**のようになっていることが多く，コンパイラはそれを1行単位で読み込みます。読み込まれた1行は**図3-3**のようにメモリに記憶します。メモリ上にソース・プログラムがある場合も，同じように1行分を別の領域へコピーしたり，直接アクセスしたりします。

ソース・プログラムの読み込みの部分を一つのサブルーチンとしておくと，ソース・プログラムの形式が異なる他のパソコンに移植するときもこの読み込みサブルーチンだけを修正するだけでよいことになります。

図3-2 テキスト・ファイルの形式

●ファイルはシーケンシャル・ファイル
●文字のコードはJISコードまたはASCII
●行の区切り EOL (End of Line) には CR, LF (16進で0D, 0A) の2文字が使われている。
●ファイルの終わり EOF (End of file) には SUB (16進で1A) の1文字が使われている。

図3-3 ソース・プログラム1行分の入力バッファ形式

3-2-2 字句の解析

読み込まれたソース・プログラムは字句解析をした後，次の構文解析に必要なトークンを求めます。

トークンには，変数名やサブルーチン名などの名前，数値定数や文字（または文字列）定数などの定数，コロン(:)やカンマ(,)，演算子(+－＊／)などの区切り記号，そしてあらかじめ意味が決められている予約語といったものがあります。

予約語とはBASIC言語でいえば“PRINT”や“INPUT”といった語で，命令や特定の値を表します。ですから，予約語になっている語は変数名やサブルーチン名などの名前には使えません。もし誤って名前のつもりで使っても，コンパイラは予約語として解釈してしまいます。

プログラム言語の種類によっては（たとえばFORTRANやPL/Iなど）予約語がないものもあります。こ

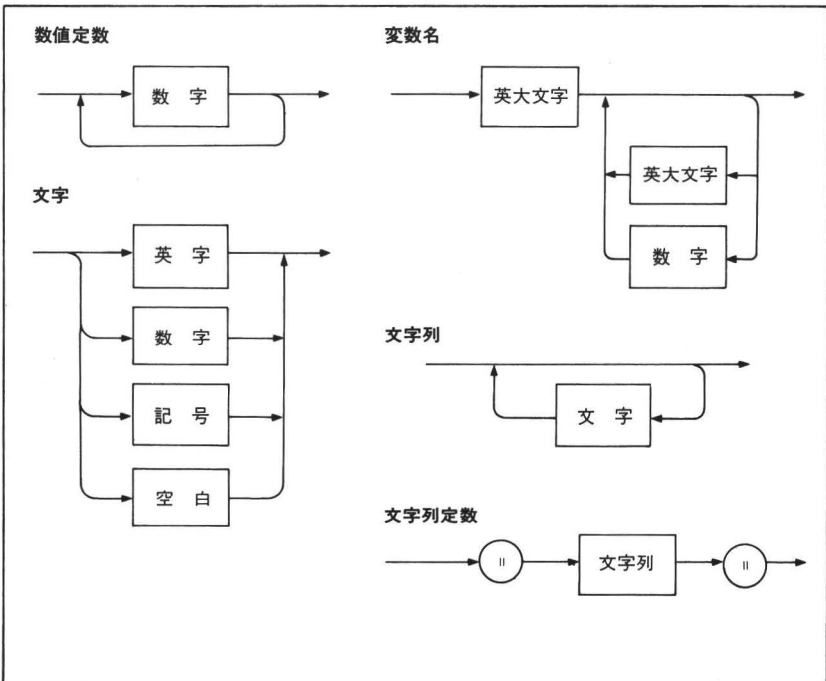
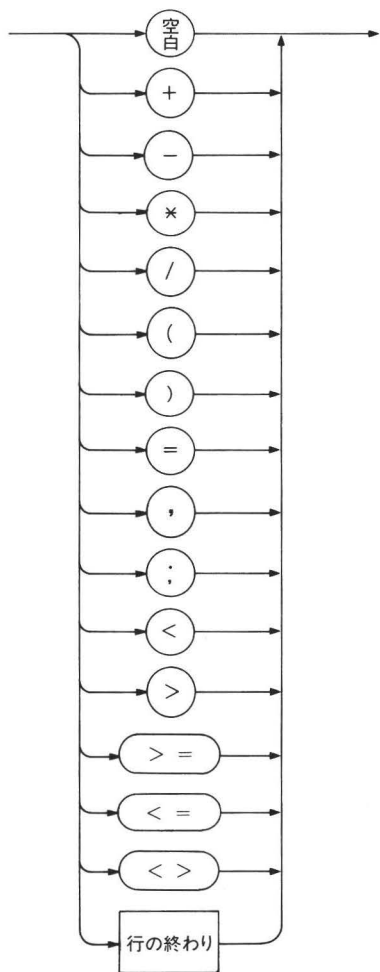
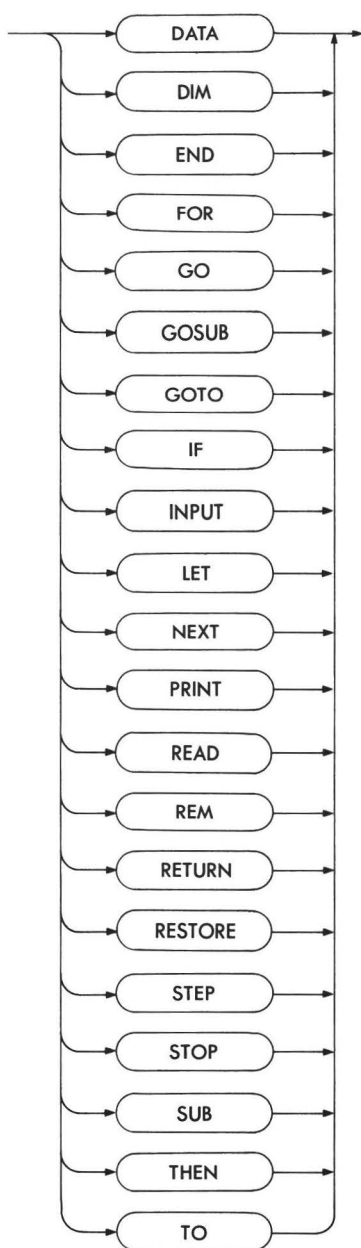
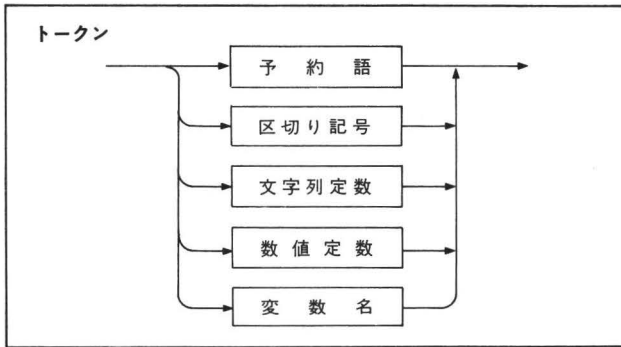
のような言語ではトークンの並びかただけで構文を判断します。たとえばPL/Iでは

```
IF IF=0 THEN IF=THEN*2;
```

といった文も書けます。しかし，予約語のない言語はコンパイラの構文解析部が複雑になります。本書では以後，予約語を持っているものとして説明していきます。

字句解析をするためには，まずトークンには何があるかを決めねばなりません。そこで，対象のプログラム言語の文法書から関係する構文を抜き出すことから始めます。これからトークンを求めるサブルーチンを設計するわけです。

ここで仮想的なBASIC言語(整数演算だけの，いわゆるTiny BASIC) について，字句解析に必要な構文を抜き出した結果，次のようになったとします。

区切り記号
空白
+
−
*
/
(
)
=
,
;
<
>
> =
< =
< >
行の終わり
予約語
DATA
DIM
END
FOR
GO
GOSUB
GOTO
IF
INPUT
LET
NEXT
PRINT
READ
REM
RETURN
RESTORE
STEP
STOP
SUB
THEN
TO

次に，これらのうちでトークンとして構文解析へ渡すべき構文を決めます。この例の場合

となります。

このあとは，文法上の制限を加味します。この仮想BASICの構文で注意しなければならないのは，区切り記号のうちの“空白”と“行の終わり”です。区切り記号の“空白”は純粋にトークンとトークンの区切りだけに使われるのに，他の区切り記号，たとえば＋－＊／などは区切り以外に演算子としての意味も持っています。この空白は構文解析に渡しても意味がないので，字句解析をしたら捨ててしまうようにします。区切り記号としての“行の終わり”は構文解析部で行番号を識別するために必要になりますが，もし構文解析で必要としないのなら空白と同じ扱いをします。

このBASIC言語の字句解析をするプログラムを**リスト3-1**に示します。これは，NECのPC-9801のN88-日本語BASIC(86)でつくりました。キーボードから入力した1行の文字列をソース・プログラムとして字句解析し，トークンそのものとその種類を表示します。予約語“END”を含む文字列がキーボードから入力されるまで，この動作を繰り返します（**実行例3-1**参照）。

●リスト3-1

```
100 '***********************************
110 '
120 '     仮想BASICの字句解析プログラム
130 '
140 '***********************************
150 '
160 DEFINT A-Z
200 '    -----  初期化  -----
210 EOL$=CHR$(0)
220 CHAR$=EOL$
230 TOKEN.CODE=0
300 '    -----  トークンの表示  -----
310 PRINT
320 PRINT "----------< Start >----------"
330 WHILE TOKEN.CODE<>52
340   GOSUB *GET.TOKEN
350   PRINT USING " ｺｰﾄﾞ : ##     ﾄｰｸﾝ : @";TOKEN.CODE,TOKEN$
360 WEND
370 PRINT "----------<  End  >----------"
380 END
1000 '------------------------------
1010 '      字句解析サブルーチン
1020 '------------------------------
1030 *GET.TOKEN
1040 '  -----  先行する空白を取る  -----
1050 WHILE CHAR$=" "
1060   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1070 WEND
1080 '  -----  行の終り  -----
1090 IF NOT(CHAR$=EOL$) THEN 1160
1100   GOSUB *READ.SOU
1110   CHAR$=MID$(SOU.LINE$,1,1): CP=2
1120   TOKEN.CODE=0
1130   TOKEN$=""
1140   RETURN
1150 '  -----  変数名 および 予約語  -----
1160 IF NOT(CHAR$>="A" AND CHAR$<="Z") THEN 1340
1170   TOKEN.CODE=1
1180   TOKEN$=""
1190   WHILE (CHAR$>="A" AND CHAR$<="Z") OR (CHAR$>="0" AND CHAR$<="9")
1200     TOKEN$=TOKEN$+CHAR$
1210     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1220   WEND
1230   RESTORE 1290: READ N
1240   FOR I=1 TO N
1250     READ RES.WORD$
1260     IF TOKEN$=RES.WORD$ THEN TOKEN.CODE=I+49: RETURN
1270   NEXT I
1280   RETURN
1290     DATA 21
1300     DATA DATA,DIM,END,FOR,GO,GOSUB,GOTO,IF,INPUT,LET
1310     DATA NEXT,PRINT,READ,REM,RESTORE,RETURN,STEP,STOP,SUB,THEN
1320     DATA TO
1330 '  -----  数値定数  -----
1340 IF NOT(CHAR$>="0" AND CHAR$<="9") THEN 1430
1350   TOKEN.CODE=2
1360   TOKEN$=""
1370   WHILE CHAR$>="0" AND CHAR$<="9"
1380     TOKEN$=TOKEN$+CHAR$
1390     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1400   WEND
1410   RETURN
1420 '  -----  文字列定数  ----
1430 IF NOT(CHAR$=CHR$(&H22)) THEN 1550
1440   TOKEN.CODE=3
1450   TOKEN$=""
1460   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
```

```
1470    WHILE CHAR$<>CHR$(&H22)
1480      IF NOT(CHAR$>=" " AND CHAR$<="_") THEN *TOKEN.ERROR
1490      TOKEN$=TOKEN$+CHAR$
1500      CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1510    WEND
1520    CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1530    RETURN
1540 '  -----  区切り記号  -----
1550 DELIM$="+-*/()=,;<>"
1560 FOR I=1 TO LEN(DELIM$)
1570   IF CHAR$=MID$(DELIM$,I,1) THEN 1600
1580 NEXT I
1590 GOTO *TOKEN.ERROR
1600 TOKEN.CODE=9+I
1610 TOKEN$=CHAR$
1620 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1630 IF TOKEN$=">" AND CHAR$="=" THEN TOKEN.CODE=21: GOTO 1670
1640 IF TOKEN$="<" AND CHAR$="=" THEN TOKEN.CODE=22: GOTO 1670
1650 IF TOKEN$="<" AND CHAR$=">" THEN TOKEN.CODE=23: GOTO 1670
1660 RETURN
1670 TOKEN$=TOKEN$+CHAR$
1680 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
1690 RETURN
1700 '  -----  エラー  -----
1710 *TOKEN.ERROR
1720     PRINT LEFT$(SOU.LINE$,LEN(SOU.LINE$)-1)
1730     PRINT TAB(CP-2);"^ Error"
1740     END
2000 '-----------------------------------------
2010 '        ソース・プログラムより1行入力
2020 '-----------------------------------------
2030 *READ.SOU
2040 LINE INPUT "*",SOU.LINE$
2050 SOU.LINE$=SOU.LINE$+EOL$
2060 RETURN
```

●実行例3-1

```
----------< Start >----------
*100 INPUT A7,BTG,CV002
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 100
 ｺｰﾄﾞ : 58    ﾄｰｸﾝ : INPUT
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : A7
 ｺｰﾄﾞ : 17    ﾄｰｸﾝ : ,
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : BTG
 ｺｰﾄﾞ : 17    ﾄｰｸﾝ : ,
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : CV002
*110 FOR I=1 TO A7 STEP BTG
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 110
 ｺｰﾄﾞ : 53    ﾄｰｸﾝ : FOR
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : I
 ｺｰﾄﾞ : 16    ﾄｰｸﾝ : =
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 1
 ｺｰﾄﾞ : 70    ﾄｰｸﾝ : TO
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : A7
 ｺｰﾄﾞ : 66    ﾄｰｸﾝ : STEP
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : BTG
*120   IF I>=CV002 THEN 140
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 120
 ｺｰﾄﾞ : 57    ﾄｰｸﾝ : IF
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : I
 ｺｰﾄﾞ : 21    ﾄｰｸﾝ : >=
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : CV002
 ｺｰﾄﾞ : 69    ﾄｰｸﾝ : THEN
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 140
*130      X = X*I
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 130
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : X
 ｺｰﾄﾞ : 16    ﾄｰｸﾝ : =
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : X
 ｺｰﾄﾞ : 12    ﾄｰｸﾝ : *
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : I
*140 NEXT I
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 140
 ｺｰﾄﾞ : 60    ﾄｰｸﾝ : NEXT
 ｺｰﾄﾞ :  1    ﾄｰｸﾝ : I
*150 END
 ｺｰﾄﾞ :  0    ﾄｰｸﾝ :
 ｺｰﾄﾞ :  2    ﾄｰｸﾝ : 150
 ｺｰﾄﾞ : 52    ﾄｰｸﾝ : END
----------<  End  >----------
```

行番号200～230が初期化の部分で，変数EOL＄，CHAR＄，TOKEN. CODEに初期値を与えています。EOL＄は行の終わりを識別するための文字で，ここではNULコード(00)としています。CHAR＄はソース・プログラムから入力した１行分の文字列のうち，次に処理する１文字を記憶するものです。最初の字句解析でソース・プログラムの最初の１行を入力するため，初期値はEOL＄と同じ文字になっています。

行番号300～380は，字句解析サブルーチン(ラベルGET. TOKEN，行番号1030～)を呼び出し，得られた解析結果を表示する部分です。字句解析サブルーチンで予約語“END”（トークン・コードは52）が得られるまで呼び出しと表示が行われます。

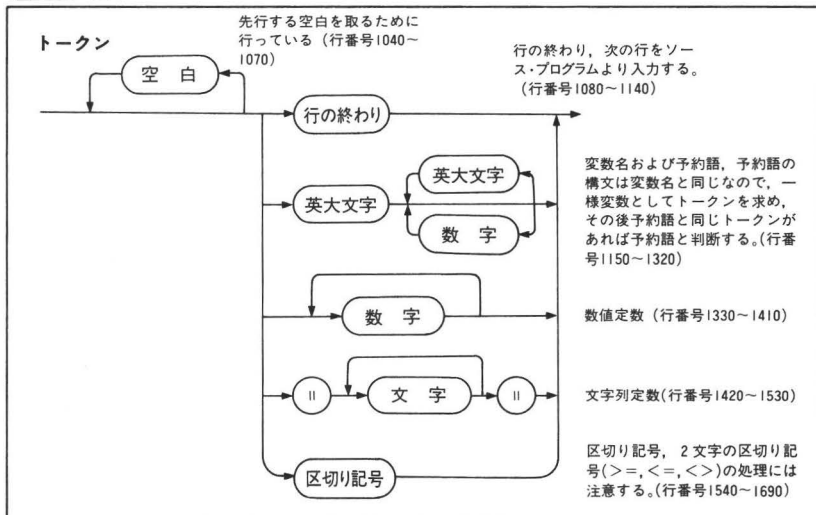
行番号1000～1740までが字句解析サブルーチンです。このサブルーチンは１回の呼び出しで１つのトークンを取り出すようになっています。そして，取り出されたトークンはTOKEN. CODEとTOKEN＄の２つの変数に入れられてきます。TOKEN＄には取り出されたトークンの文字列そのものが入り，TOKEN. CODEにはその種類がコードとして入ります。コードは**表3-1**のようになっています。このルーチンは**図3-4**のような構文図をもとにプログラミングしたものです。

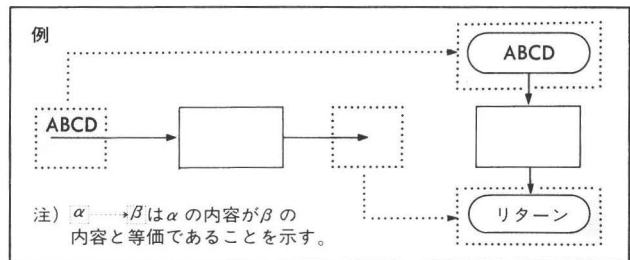
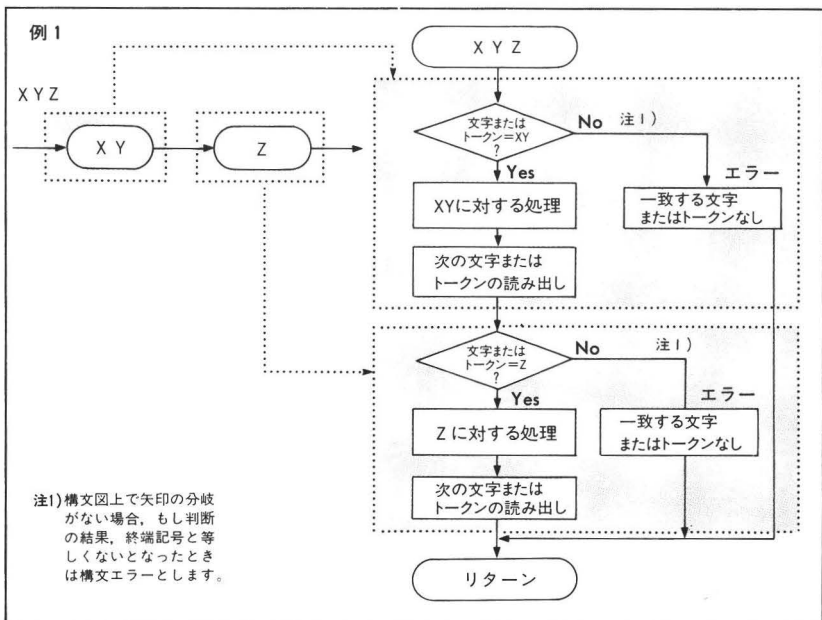
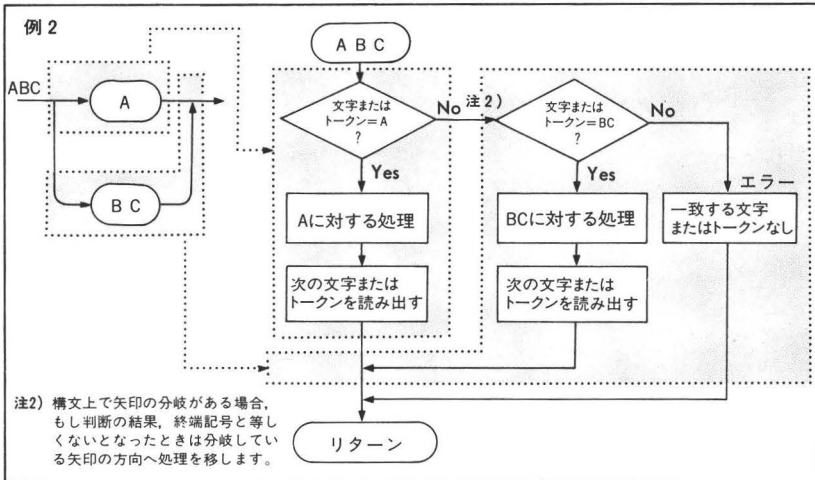
行番号2000～2060がソース・プログラムを１行入力するサブルーチン(ラベルREAD. SOU，行番号2030）で，キーボードから入力された１行分の文字列は最後にEOL＄がつけ加えられ，変数SOU. LINE＄に入れられてきます。

表3-1　仮想 BASICのトークン・コード

コード	トークン
0	行の終わり
1	変数名
2	数値定数
3	文字列定数
10	＋
11	−
12	＊
13	／
14	(
15	)
16	＝
17	,
18	;
19	<
20	>
21	>=
22	<=
23	<>
50	DATA
51	DIM
52	END
53	FOR
54	GO
55	GOSUB
56	GOTO
57	IF
58	INPUT
59	LET
60	NEXT
61	PRINT
62	READ
63	REM
64	RESTORE
65	RETURN
66	STEP
67	STOP
68	SUB
69	THEN
70	TO

※ コード 10～23が区切り記号
　コード 50～70が予約語

図3-4　仮想 BASICのトークンの構文図

さて，字句解析のルーチンは**図3-4**の構文図そのままになっていることがわかると思います。それは，構文図がコンパイラ作成時の一種の概略流れ図として使えるということです。構文図から字句解析や構文解析の詳細流れ図，およびプログラムをつくるときの規則をまとめておきます。

①構文図の入口と出口は，その構文をプログラムにしたときサブルーチンのエントリーとリターンになります。

② 〈終端記号〉は，終端記号に対する判断となります。

③〈非終端記号〉は，非終端記号に相当するサブルーチンの呼び出しになります。

例 1

S T U V

S T

U V

S T U V

S T

サブルーチン
STは正しく処理
されたか?

No

注3)

エラー

サブルーチンSTUVは正しく処理されなかった

Yes

S Tに対する処理

U V

サブルーチン
UVは正しく処理
されたか？

No

注3)

Yes

エラー

U Vに対する処理

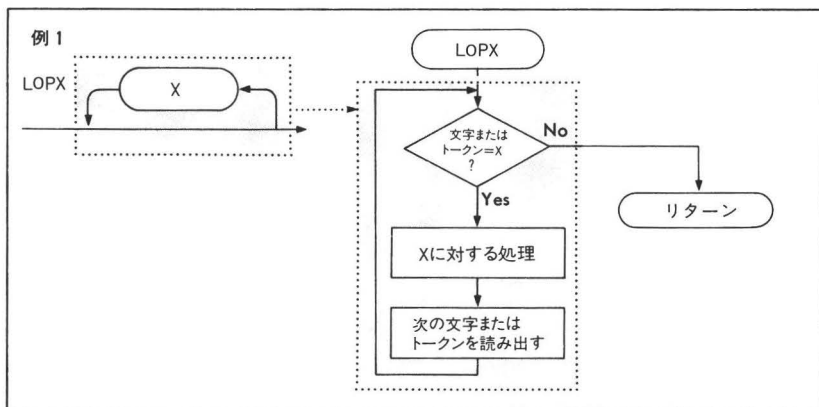
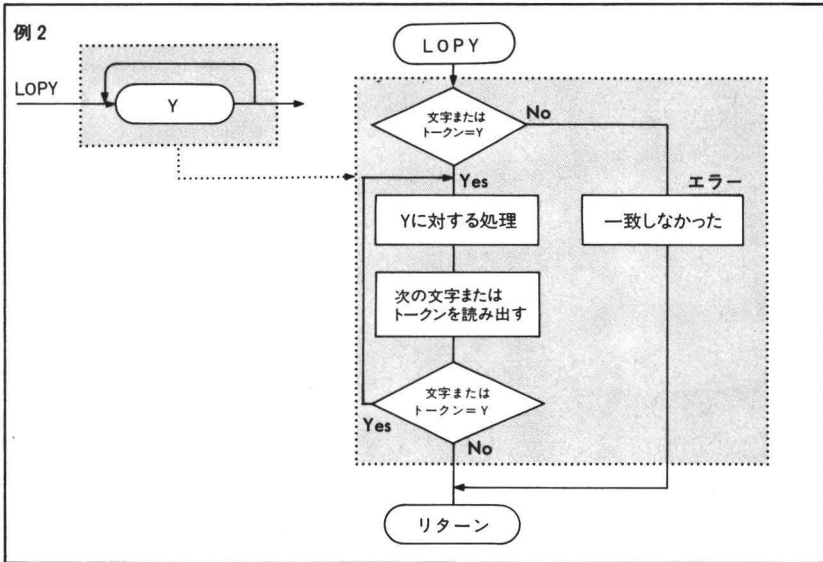
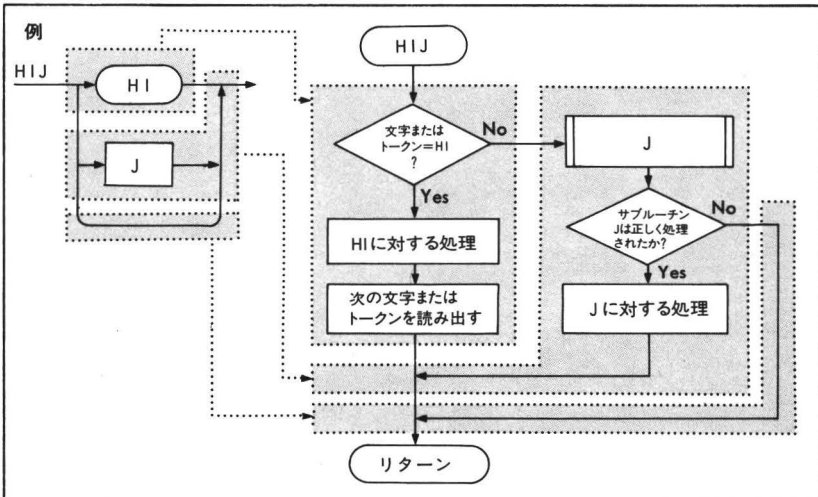
サブルーチンSTUVは正しく処理されなかった

リターン

注3）構文上で分岐がない場合，相当するサブルーチンが正しく実行されなかったとき（構文が正しくなかったとき）は構文エラーとします。しかし，サブルーチンから正しく実行されたかどうかが渡されない場合，この判断自体が必要ありません。

例 2

DEFG
DE
FG
DEFG
DE
No
サブルーチンDEは正しく処理されたか?
Yes
DEに対する処理
FG
サブルーチンFGは正しく処理されたか?
No
Yes
エラー
FGに対する処理
サブルーチンDEFGは正しく処理されなかった
リターン

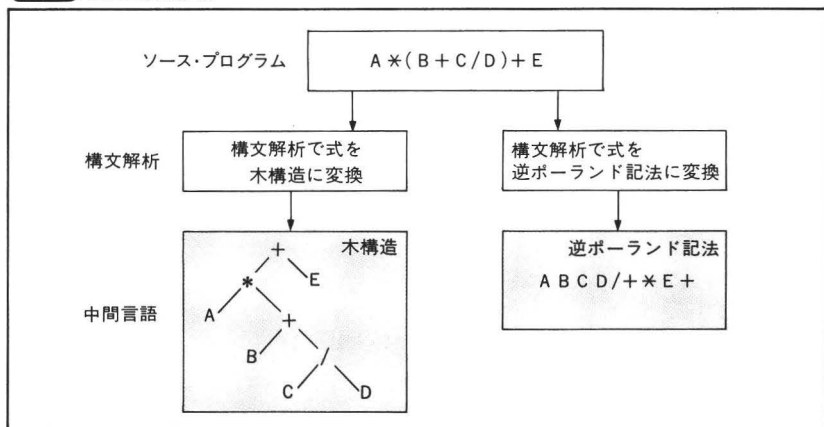
④矢印が前に戻り，繰り返しになっている構文はプログラムも繰り返しにします。

⑤いままでの例では，構文図で矢印の先の終端記号と等しいものがなかったり，非終端記号に相当するサブルーチンを実行しても正しく処理されなかった場合などをエラーとしていましたが，分岐が出口まで通じているときはエラーの処理をしません。

3-3 式のコンパイル

コンパイラの作成というとまず式のことが問題にされるくらい，式のコンパイルは重要です。この項では，代入文やＩＦ文の中に現われる式をどのようにすれば機械語（オブジェクト・プログラム）に変換できるのかについて述べます。まず式の構文解析の諸方法について解説し，最適化，コード生成についても触れます。

3-3-1 式の構文解析の方法

構文解析には二通りの方法があります。第一の方法は，字句解析部で読み込まれた式を構文解析部でいったん木構造（tree structure）や逆ポーランド記法のような中間言語に変換して最適化し，コード生成部ではこの中間言語からオブジェクト・プログラムをつくり出す方法です。

第二の方法は，構文解析のプログラム自体を木構造にする方法です。この方法はコンパイラを再帰的につくることで実現できます。この場合構文図で書かれた構文を

図3-5 式の構文解析

そのままプログラムにします。もともと構文図は文法を表すのに再帰的な定義を使っているからです。この方法は構文解析部が直接，最適化，コード生成を呼び出しオブジェクト・プログラムをつくります。

以降これら二通りについて，例を挙げながら解説していきます。ただし，木構造の中間言語をつくる方式は他に比べて難しいため，割愛しました。

3-3-2 逆ポーランド記法を用いて式をコンパイルする

逆ポーランド記法はポーランドの数学者カジヴィッツが提案した記法です。普通の式が

変数1　演算子　変数2　(例．A+B)

と書くところを

変数1　変数2　演算子　(例．AB+)

と書きます。“A＋B”を日本語で“AにBを加える”というのに似ています。

もっと複雑な式も逆ポーランド記法にできます。たとえば

A+B／(C−2)+D＊E

という式があるとします。ここで，各演算子を“二つの引数を持つ関数”と考えて“2＋3”を（2，3)＋のように表すと

((A，(B，(C，2) −) ／)+，(D，E)＊) +

となります。ここからカンマと（　）を取り除くと

ABC2−／+DE＊+

となり逆ポーランド記法になります。

逆ポーランド記法の他にもポーランド記法という記法があるのですが，コンパイラでは逆ポーランド記法のほうがよく使われるので，逆ポーランド記法のことを単にポーランド記法と呼ぶことが多いようです。

では，この変換をするプログラムのアルゴリズムを説明します。

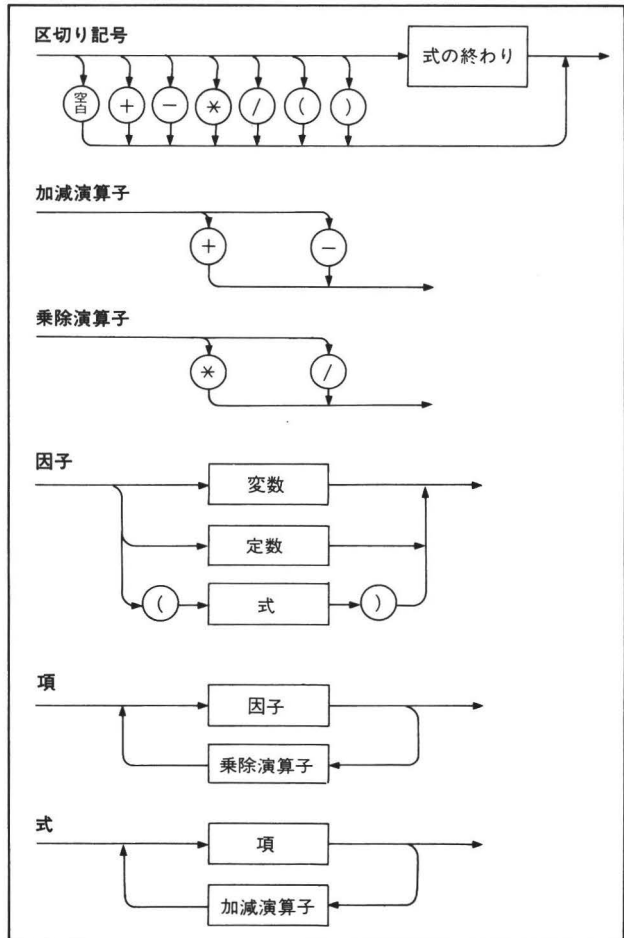
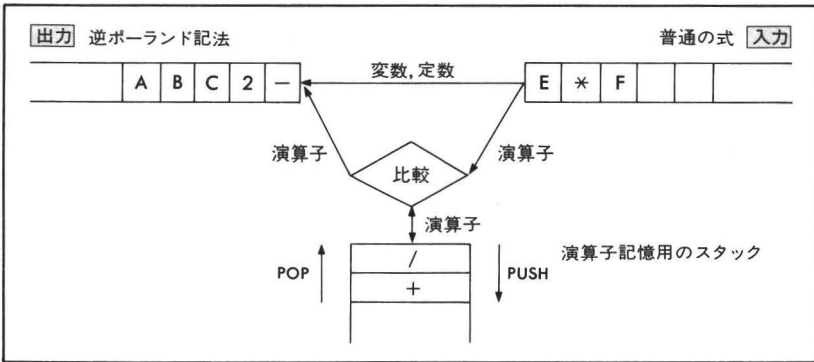
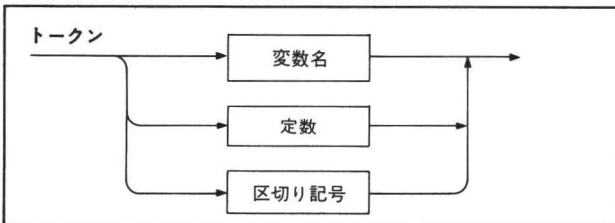
初めにここで例として使う式の構文を定義しておきましょう。

普通の式を逆ポーランド記法に変換するときまず考えなければならないことは，“演算子の優先順位”の処理です。もし演算子間に優先順位がないなら，“A－B＊C”は“AB－C＊”となります。しかし実際には優先順位があり，これは“ABC＊－”とするべきです。そこで優先順位の処理にスタックを使います。そして変換のとき，入力された演算子とスタック・トップの演算子とを比較して優先順位を考慮します。逆ポーランド記法への変換の概要を図にすると**図3-6**のようになります。

図3-6　逆ポーランド記法への変換の概要

次に考えなければならないのは，トークンの出現順序です。ここで例に使っている式のトークンは

で，区切り記号としての空白はトークンとはしません。この式のトークンの出現順序は，構文図から次のように考えられます。

①式は変数名，定数，“(”のいずれかから始まる。
②変数名，定数，“)”の次には，加減演算子，乗除演算子，“)”のいずれかがくる。
③加減演算子，乗除演算子，“(”の次には，変数名，定数，“(”のいずれかがくる。

変換する式がこの出現順序にしたがわない場合，エラーとします。

以上のことから，式を逆ポーランド記法に変換するアルゴリズムは**図3-7**の流れ図のようになります。この流れ図を使って式“A＋B／(C－2)＋D＊E”を逆ポーランド記法に変換する過程を**図3-8**に示します。

次に逆ポーランド記法の式を機械語へ変換します。式を機械語に変換するとき，演算の中間結果を一時的に記憶しなければならなくなることがしばしば発生します。このような場合，スタック操作の命令(PUSH，POPなど）を使って，中間結果をスタックに積むようなオブジェクト・コードをつくります。

逆ポーランド記法の式をアセンブラ言語に変換する規則は次のようになります。ただし最適化をしない一番簡単な場合です。また，生成するコードはすべて16ビットの符号つき整数とし，オーバーフローは考えないことにします。

①逆ポーランド記法の式は左から右へトークン単位で入力されるものとする。
②トークンが変数名なら記号表（後述）からその変数のアドレスを求め，オブジェクト・コードとして

```
PUSH HL
LD   HL,(変数のアドレス)
```

を出力する。ただし，“PUSH　HL”は初めてのオブジェクト・コード生成のときには出力しない。

図3-7 逆ポーランド記法への変換アルゴリズムの流れ図

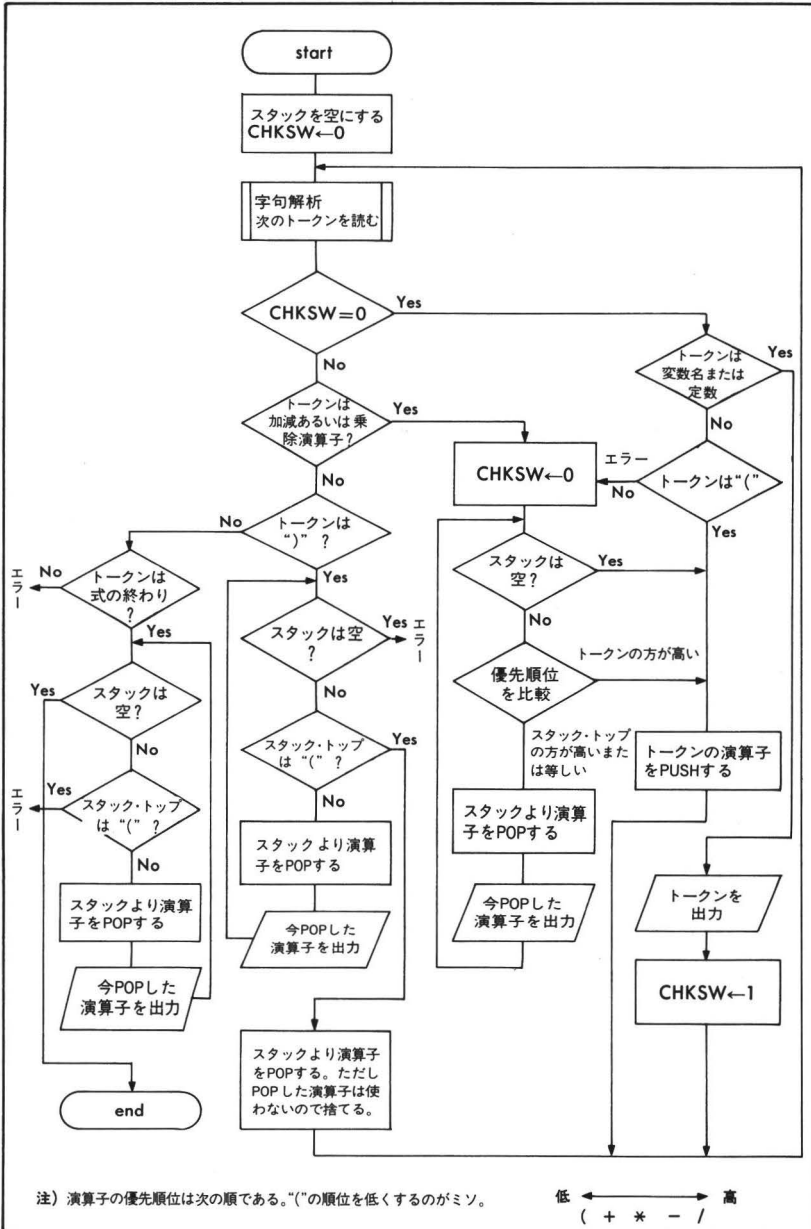



図3-8 逆ポーランド記法への変換過程

入力(トークン)	スタック	出力(逆ポーランド記法)
A		
		A
+		A
	+	A
B	+	A
	+	A B
/	+	A B
	/ +	A B
(	/ +	A B
	(/ +	A B
C	(/ +	A B
	(/ +	A B C
−	(/ +	A B C
	− (/ +	A B C
2	− (/ +	A B C
	− (/ +	A B C 2
)	− (/ +	A B C 2
	/ +	A B C 2 −
+	/ +	A B C 2 −
	+	A B C 2 − / +
D	+	A B C 2 − / +
	+	A B C 2 − / + D
*	+	A B C 2 − / + D
	* +	A B C 2 − / + D
E	* +	A B C 2 − / + D
	* +	A B C 2 − / + D E
式の終わり	* +	A B C 2 − / + D E
		A B C 2 − / + D E * +

③トークンが定数ならオブジェクト・コードとして

```
PUSH HL
LD   HL, 定数
```

を出力する。ただし“PUSH HL”は初めてのオブジェクト・コード生成のときには出力しない。また，定数は 0 ～32767以外の値ならエラーとする。

④トークンが演算子＋－＊／なら次のようなオブジェクト・コードを生成する。

●加算（＋）

```
POP  DE     ;HLの値にスタック・トップの
ADD  HL,DE  ;値を加算し，HLへ入れる
```

●減算（―）

```
EX   DE,HL  ;
POP  HL     ;スタック・トップの値よりHL
OR   A      ;の値を減算し，HLへ入れる
SBC  HL,DE  ;
```

●乗算（＊）

```
POP   DE    ;HLの値にスタック・トップ
CALL IMUL   ;の値を乗算し，HLに入れる。
            ;乗算(HL←HL＊DE)はサブ
            ;ルーチンIMULが行う。
```

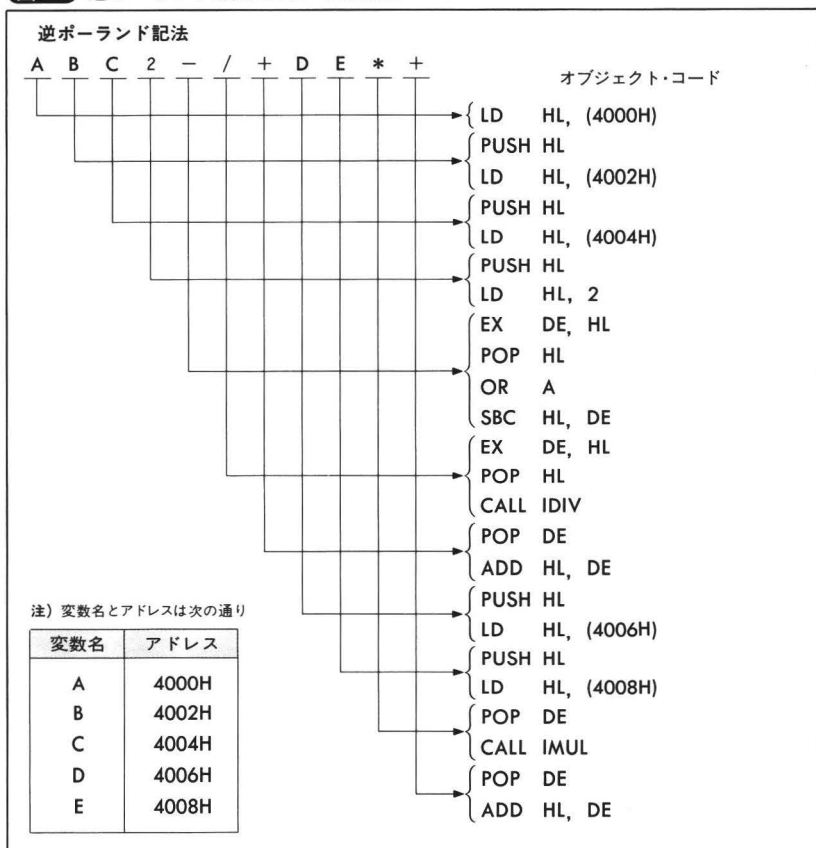
●除算（／）

```
EX    DE,HL ;HLの値をスタック・トップ
POP   HL    ;の値で除算し，HLに入れる。
CALL  IDIV  ;除算（HL←HL／DE）はサ
            ;ブルーチンIDIVが行う。
```

⑤逆ポーランド記法の式にミスがなければ，ＨＬレジスタに演算結果が入るような機械語（アセンブラ言語）のオブジェクト・コードがつくられる。

以上のことから，たとえば逆ポーランド記法の式“ABC2－/＋DE ＊＋”を機械語(アセンブラ言語)に変換すると，**図3-9**のようになります。

リスト3-2は，これまでの説明をもとに，キーボードから入力された式をいったん逆ポーランド記法に変換し，続いてＺ80用の機械語（アセンブラ言語）に変換するプログラム（N88―BASＩC（86）で作成）です（**実行例**

図3-9 逆ポーランド記法の式から機械語への変換例

3-2参照)。このプログラムでは，オブジェクト・プログラムが16進数で1000番地から，変数は4000番地から始まるようになっています。そして，ランタイム・ルーチンとして3000番地に乗算（HL ← HL * DE），3003番地に除算（HL ← HL/DE）ルーチンがあるものとします。

行番号1000〜1430が普通の式を逆ポーランド記法に変換する部分で，コンパイラの構文解析に当たります。行番号1440〜2060が逆ポーランド記法の式からオブジェクト・コードを生成する部分で，コンパイラのコード生成に当たる部分です。

●リスト3-2

```
100 '************************************
110 '
120 '       逆ポーランド記法を用いた式のコンパイル
130 '
140 '          (Z80  CPUの機械語に変換)
150 '
160 '************************************
170 '
180 DEFINT A-Z
190 DIM POL$(100),OPRSTK$(50),SYMTBL$(20)
200 DEF FNH2$(X!)=RIGHT$("0"+HEX$(X!),2)
210 DEF FNH4$(X!)=RIGHT$("000"+HEX$(X!),4)
220 DEF FNRH4$(X!)=RIGHT$(FNH4$(X!),2)+LEFT$(FNH4$(X!),2)
230 '
240 EOL$=CHR$(0)
250 GOSUB *READ.SOU
1000 '========================================
1010       普通の式を逆ポーランド記法の式に変換
1020 '========================================
1030 PRINT
1040 PRINT "-----<  普通の式を逆ポーランド記法の式に変換  >-----"
1050 PRINT
1060 PRINT "入力された式    : ";IN.EXPR$
1070 PPTR=0: SPTR=1
1080 OPRSTK$(0)=" "
1090 CHKSW=0: LFLG=-1
1100 WHILE LFLG
1110   GOSUB *GET.TOKEN
1120   IF CHKSW<>0 THEN 1200
1130     IF NOT(TOKEN.CODE=1 OR TOKEN.CODE=2) THEN 1170
1140       POL$(PPTR)=CHR$(TOKEN.CODE)+TOKEN$: PPTR=PPTR+1
1150       CHKSW=1
1160     GOTO 1330
1170       IF TOKEN$<>"(" THEN *SYNTAX.ERROR
1180       OPRSTK$(SPTR)=CHR$(TOKEN.CODE)+TOKEN$: SPTR=SPTR+1
1190     GOTO 1330
1200   IF NOT(TOKEN.CODE>=12 AND TOKEN.CODE<=15) THEN 1270
1210     CHKSW=0
1220     WHILE SPTR<>1 AND TOKEN.CODE¥2<=ASC(OPRSTK$(SPTR-1))¥2
1230       POL$(PPTR)=OPRSTK$(SPTR-1): PPTR=PPTR+1: SPTR=SPTR-1
1240     WEND
1250     OPRSTK$(SPTR)=CHR$(TOKEN.CODE)+TOKEN$: SPTR=SPTR+1
1260     GOTO 1330
1270   IF NOT(TOKEN$=")") THEN LFLG=0: GOTO 1330
1280     WHILE SPTR<>1 AND MID$(OPRSTK$(SPTR-1),2)<>"("
1290       POL$(PPTR)=OPRSTK$(SPTR-1): PPTR=PPTR+1: SPTR=SPTR-1
1300     WEND
1310     IF SPTR=1 THEN *SYNTAX.ERROR
1320     SPTR=SPTR-1
1330 WEND
1340 IF TOKEN.CODE<>0 THEN *SYNTAX.ERROR
1350 WHILE SPTR<>1
1360   IF MID$(OPRSTK$(SPTR-1),2)="(" THEN *SYNTAX.ERROR
1370   POL$(PPTR)=OPRSTK$(SPTR-1):, PPTR=PPTR+1: SPTR=SPTR-1
1380 WEND
1390 PRINT "逆ポーランド記法 : ";
1400 FOR I=0 TO PPTR-1
1410   PRINT MID$(POL$(I),2);" ";
1420 NEXT
1430 PRINT
1440 '==========================================
1450 '     逆ポーランド記法の式よりオブジェクト・コードを生成
1460 '==========================================
1470 PRINT
1480 PRINT "-----<  逆ポーランド記法の式よりオブジェクト・コードを生成  >-----"
1490 PRINT
1500 PRINT "  **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **"
```

```
1510 PRINT
1520 SYMTBL$(0)="$"
1530 LOCADR=&H1000
1540 PSHOTF=0
1550 FOR I=0 TO PPTR-1
1560   TOKEN.CODE=ASC(POL$(I)): TOKEN$=MID$(POL$(I),2)
1570   PRINT FNH4$(LOCADR);" ";
1580   IF NOT(TOKEN.CODE=1) THEN 1650
1590     GOSUB *SYMTBL.SEA
1600     GOSUB *PUSHOUT                        ' PUSH HL
1610     PRINT "2A";FNRH4$(VARADR);            ' LD HL,( 変数のアドレス )
1620     PRINT TAB(15);"LD      HL,(";FNH4$(VARADR);"H)"
1630     LOCADR=LOCADR+3
1640     GOTO 2060
1650   IF NOT(TOKEN.CODE=2) THEN 1730
1660     CONS!=VAL(TOKEN$)
1670     IF CONS!>&H7FFF THEN *GEN.ERROR
1680     GOSUB *PUSHOUT                        ' PUSH HL
1690     PRINT "21";FNRH4$(CONS!);             ' LD HL,定数
1700     PRINT TAB(15);"LD      HL,";TOKEN$
1710     LOCADR=LOCADR+3
1720     GOTO 2060
1730   IF NOT(TOKEN$="+") THEN 1800
1740     PRINT "D1";                           ' POP DE
1750     PRINT TAB(15);"POP     DE"
1760     PRINT FNH4$(LOCADR+1);" 19";          ' ADD HL,DE ——
1770     PRINT TAB(15);"ADD     HL,DE"
1780     LOCADR=LOCADR+2
1790     GOTO 2060
1800   IF NOT(TOKEN$="-") THEN 1910
1810     PRINT "EB";                           ' EX DE,HL
1820     PRINT TAB(15);"EX      DE,HL"
1830     PRINT FNH4$(LOCADR+1);" E1";          ' POP HL
1840     PRINT TAB(15);"POP     HL"
1850     PRINT FNH4$(LOCADR+2);" B7";          ' OR A
1860     PRINT TAB(15);"OR      A"
1870     PRINT FNH4$(LOCADR+3);" ED52";        ' SBC HL,DE
1880     PRINT TAB(15);"SBC     HL,DE"
1890     LOCADR=LOCADR+5
1900     GOTO 2060
1910   IF NOT(TOKEN$="*") THEN 1980
1920     PRINT "D1";                           ' POP DE
1930     PRINT TAB(15);"POP     DE"
1940     PRINT FNH4$(LOCADR+1);" CD0030";      ' CALL IMUL
1950     PRINT TAB(15);"CALL    IMUL"
1960     LOCADR=LOCADR+4
1970     GOTO 2060
1980   IF NOT(TOKEN$="/") THEN *GEN.ERROR
1990     PRINT "EB";                           ' EX DE,HL
2000     PRINT TAB(15);"EX      DE,HL"
2010     PRINT FNH4$(LOCADR+1);" E1";          ' POP HL
2020     PRINT TAB(15);"POP     HL"
2030     PRINT FNH4$(LOCADR+2);" CD0330";      ' CALL IDIV
2040     PRINT TAB(15);"CALL    IDIV"
2050     LOCADR=LOCADR+5
2060 NEXT I
2070 '
2080 PRINT
2090 PRINT "  **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **"
2100 PRINT
2110 FOR I=0 TO 19
2120   IF SYMTBL$(I)="$" THEN 2160
2130   PRINT FNH4$(&H4000+I*2);" ";SYMTBL$(I)
2140 NEXT I
2150 PRINT
2160 END
2170 '----------  " PUSH HL " の出力  ----------
2180 *PUSHOUT
2190 IF PSHOTF=0 THEN PSHOTF=1: RETURN
```

```
2200 PRINT "E5";                                        ' PUSH HL
2210 PRINT TAB(15);"PUSH    HL"
2220 LOCADR=LOCADR+1
2230 PRINT FNH4$(LOCADR);" ";
2240 RETURN
2250 '----------  記号表の検索および登録  ----------
2260 *SYMTBL.SEA
2270 FOR J=0 TO 19
2280   IF SYMTBL$(J)="$" THEN 2330
2290   IF SYMTBL$(J)=TOKEN$ THEN 2350
2300 NEXT J
2310 PRINT: PRINT "SYMBOL TABLE OVERFLOW"
2320 END
2330 SYMTBL$(J)=TOKEN$
2340 SYMTBL$(J+1)="$"
2350 VARADR=&H4000+J*2
2360 RETURN
2370 '----------  構文 エラー  ----------
2380 *SYNTAX.ERROR
2390 PRINT: PRINT "SYNTAX ERROR"
2400 END
2410 '----------  コード生成 エラー  ----------
2420 *GEN.ERROR
2430 PRINT: PRINT "CODE GENERATION ERROR"
2440 END
5000 '==============================
5010 '      字句解析サブルーチン
5020 '==============================
5030 *GET.TOKEN
5040 '  -----  先行する空白を取る  -----
5050 WHILE CHAR$=" "
5060   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5070 WEND
5080 '  -----  行の終り  -----
5090 IF NOT(CHAR$=EOL$) THEN 5140
5100   TOKEN.CODE=0
5110   TOKEN$=""
5120   RETURN
5130 '  -----  変数名  -----
5140 IF NOT(CHAR$>="A" AND CHAR$<="Z") THEN 5230
5150   TOKEN.CODE=1
5160   TOKEN$=""
5170   WHILE (CHAR$>="A" AND CHAR$<="Z") OR (CHAR$>="0" AND CHAR$<="9")
5180     TOKEN$=TOKEN$+CHAR$
5190     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5200   WEND
5210   RETURN
5220 '  -----  数値定数  -----
5230 IF NOT(CHAR$>="0" AND CHAR$<="9") THEN 5320
5240   TOKEN.CODE=2
5250   TOKEN$=""
5260   WHILE CHAR$>="0" AND CHAR$<="9"
5270     TOKEN$=TOKEN$+CHAR$
5280     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5290   WEND
5300   RETURN
5310 '  -----  区切り記号  -----
5320 DELIM$="()+-*/"
5330 FOR I=1 TO LEN(DELIM$)
5340   IF CHAR$=MID$(DELIM$,I,1) THEN 5370
5350 NEXT I
5360 GOTO *TOKEN.ERROR
5370 TOKEN.CODE=9+I
5380 TOKEN$=CHAR$
5390 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5400 RETURN
```

```
5410 '  -----  エラー  -----
5420 *TOKEN.ERROR
5430     PRINT LEFT$(SOU.LINE$,LEN(SOU.LINE$)-1)
5440     PRINT TAB(CP-2);"^ Error"
5450     END
6000 '========================================
6010         ソース・プログラム(式)の入力
6020 '========================================
6030 *READ.SOU
6040 LINE INPUT "式を入力 : ",IN.EXPR$
6050 SOU.LINE$=IN.EXPR$+EOL$
6060 CHAR$=MID$(SOU.LINE$,1,1): CP=2
6070 RETURN
```

●実行例3-2

```
式を入力 : A + B/( C-2 ) + D*E

-----<  普通の式を逆ポーランド記法の式に変換  >-----

入力された式    : A + B/( C-2 ) + D*E
逆ポーランド記法 : A B C 2 - / + D E * +

-----<  逆ポーランド記法の式よりオブジェクト・コードを生成  >-----

  **  オブジェクト・プログラム  **

1000 2A0040     LD      HL,(4000H)
1003 E5         PUSH    HL
1004 2A0240     LD      HL,(4002H)
1007 E5         PUSH    HL
1008 2A0440     LD      HL,(4004H)
100B E5         PUSH    HL
100C 210200     LD      HL,2
100F EB         EX      DE,HL
1010 E1         POP     HL
1011 B7         OR      A
1012 ED52       SBC     HL,DE
1014 EB         EX      DE,HL
1015 E1         POP     HL
1016 CD0330     CALL    IDIV
1019 D1         POP     DE
101A 19         ADD     HL,DE
101B E5         PUSH    HL
101C 2A0640     LD      HL,(4006H)
101F E5         PUSH    HL
1020 2A0840     LD      HL,(4008H)
1023 D1         POP     DE
1024 CD0030     CALL    IMUL
1027 D1         POP     DE
1028 19         ADD     HL,DE

  **  シンボル・テーブル  **

4000 A
4002 B
4004 C
4006 D
4008 E
```

さて，ここまでが逆ポーランド記法を使って式をコンパイルする方法なのですが，オブジェクト・プログラムをよく見ると無駄なPUSHやPOP命令が生成されています。たとえば逆ポーランド記法で

```
A 19 B * +
```

という式は

```
LD   HL,（変数Aのアドレス）
PUSH HL
LD   HL, 19
PUSH HL
LD   HL,（変数Bのアドレス）
POP  DE
CALL IMUL
POP  DE
ADD  HL, DE
```

というオブジェクト・プログラムになりますが，

```
LD   HL,（変数Bのアドレス）
LD   DE, 19
CALL IMUL
LD   DE,（変数Aのアドレス）
ADD  HL, DE
```

とすれば無駄なPUSH, POP命令がなくなって，効率が良くなります。このようにプログラムの効率を上げるための処理が最適化です。以下では逆ポーランド記法の式をオブジェクト・コードに変換するときの，割合簡単にできる最適化について説明します。

無駄なPUSH, POP命令を生成させないないようにするためには演算の順序を変えてやります。つまり以下の規則のように，コード生成時にまだロードする必要がない変数や定数はスタックに積み，必要になったときス

タックから取り出すことにするのです。
①逆ポーランド記法の式は左から右へトークン単位で入力されるものとする。
②トークンが変数名や定数ならスタックにPUSHする。
③トークンが演算子なら，スタックの先頭から二つの変数名や定数，あるいは式の中間結果を示すマークなどをPOPし，**表3-2**を用いてオブジェクト・コードを生成する。この表は“x　y　演算子”(二項演算)のオブジェクト・コードを生成するためのもので，yは初めにスタックからＰＯＰしたもの，xは二番目にスタックからＰＯＰしたものを示す。こうしてコードを生成した後，スタックには新たな中間結果を示すマークをPUSHしておく。

以上のことを流れ図にすると**図3-10**のようになります。ここで**表3-2**は二項演算に対するオブジェクト・コードの生成を示す表として，再帰的な方法で式をコンパイルするとき（**3-3-3**）にも使います。

リスト3-3がこの最適化を取り入れた，逆ポーランド記法を用いて式をコンパイルするプログラムです（**実行例3-3**参照）。**実行例3-2**と比べるとオブジェクト・プログラムが小さくなっています。

表3-2 2項演算のオブジェクト・コード表

①加算および乗算のオブジェクト（普通の式でX＋YおよびX＊Y）

X＼Y	Y は変数名	Y は定数	Y は中間結果

X は変数名	PUSH HL ※1
	LD HL, (X)
	LD DE, (Y)
	演算命令
		PUSH HL ※1
		LD HL, (X)
		LD DE, Y
		演算命令
			LD DE, (X)
			演算命令

X は定数	PUSH HL ※1
	LD HL, (Y)
	LD DE, X
	演算命令
		PUSH HL ※1
		LD HL, X
		LD DE, Y
		演算命令
			LD DE, X
			演算命令

X は中間結果	LD DE, (Y) 演算命令	LD DE, Y 演算命令	POP DE 演算命令

加算や乗算のように普通の式で表して

X演算子Y＝Y演算子X

とできる演算の場合は左図のようなオブジェクト・コードを生成する。

●加算

演算命令 の部分には次のオブジェクトを生成する

ADD HL, DE

●乗算

演算命令 の部分には次のオブジェクトを生成する

CALL IMUL ；IMULは乗算（HL←HL＊DE）を行うランタイム・ルーチン

②減算および除算のオブジェクト（普通の式でX－YおよびX／Y）

X＼Y	Y は変数名	Y は定数	Y は中間結果

X は変数名	PUSH HL ※1
	LD HL, (X)
	LD DE, (Y)
	演算命令
		PUSH HL ※1
		LD HL, (X)
		LD DE, Y
		演算命令
			EX DE, HL
			LD HL, (X)
			演算命令

X は定数	PUSH HL ※1
	LD HL, X
	LD DE, (Y)
	演算命令
		PUSH HL ※1
		LD HL, X
		LD DE, Y
		演算命令
			EX DE, HL
			LD HL, X
			演算命令

X は中間結果	LD DE, (Y) 演算命令	LD DE, Y 演算命令	EX DE, HL POP HL 演算命令

減算や除算のように普通の式で表して

X演算子Y≠Y演算子X

●減算

演算命令 の部分には次のオブジェクトを生成する

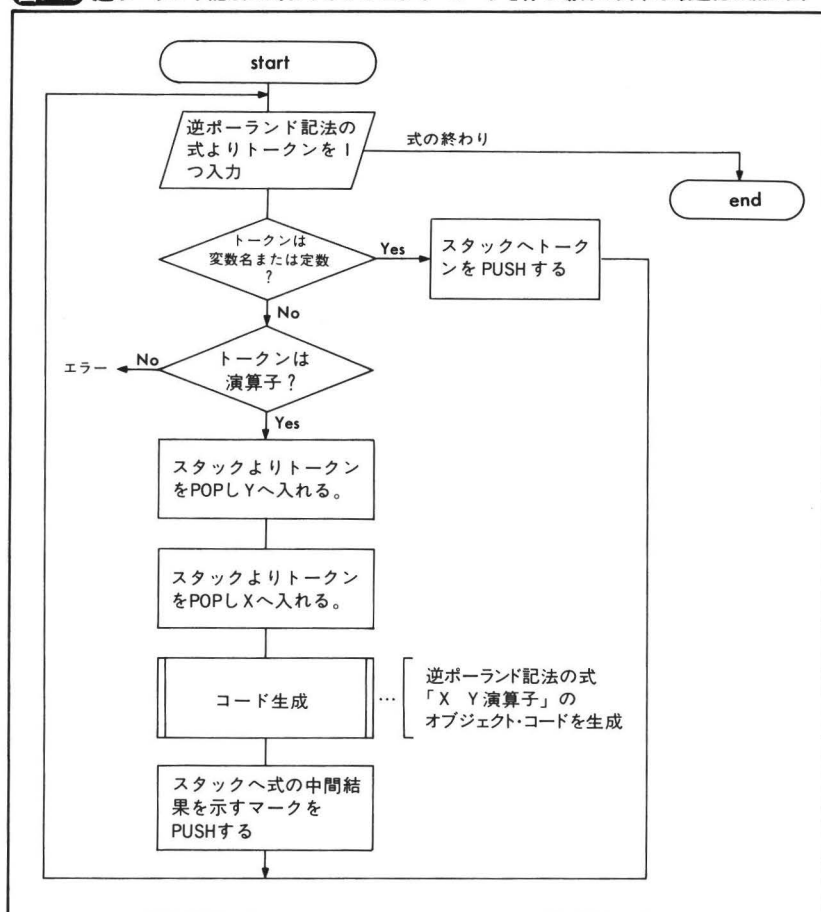
OR A
SBC HL, DE

●除算

演算命令 の部分には次のオブジェクトを生成する

CALL IDIV ；IDIVは除算（HL←HL/DE）を行うランタイム・ルーチン

※1 始めてのオブジェクト・コードの生成のときには出力しない。

図3-10 逆ポーランド記法の式よりオブジェクト・コードを作る場合の簡単な最適化の流れ図

●リスト3-3

```
100 '************************************
110 '
120 '       逆ポーランド記法を用いた式のコンパイル
130 '              簡単な最適化を行なった例
140 '
150 '           (Z80 CPUの機械語に変換)
160 '
170 '************************************
180 '
190 DEFINT A-Z
200 DIM POL$(100),OPRSTK$(50),SYMTBL$(20)
210 DEF FNH2$(X!)=RIGHT$("0"+HEX$(X!),2)
220 DEF FNH4$(X!)=RIGHT$("000"+HEX$(X!),4)
230 DEF FNRH4$(X!)=RIGHT$(FNH4$(X!),2)+LEFT$(FNH4$(X!),2)
240
250 EOL$=CHR$(0)
260 GOSUB *READ.SOU
1000 '========================================
1010 '     普通の式を逆ポーランド記法の式に変換
1020 '========================================
1030 PRINT
1040 PRINT "-----<  普通の式を逆ポーランド記法の式に変換  >-----"
1050 PRINT
1060 PRINT "入力された式     : ";IN.EXPR$
1070 PPTR=0: SPTR=1
1080 OPRSTK$(0)=" "
1090 CHKSW=0: LFLG=-1
1100 WHILE LFLG
1110   GOSUB *GET.TOKEN
1120   IF CHKSW<>0 THEN 1200
1130     IF NOT(TOKEN.CODE=1 OR TOKEN.CODE=2) THEN 1170
1140       POL$(PPTR)=CHR$(TOKEN.CODE)+TOKEN$: PPTR=PPTR+1
1150       CHKSW=1
1160     GOTO 1330
1170       IF TOKEN$<>"(" THEN *SYNTAX.ERROR
1180       OPRSTK$(SPTR)=CHR$(TOKEN.CODE)+TOKEN$: SPTR=SPTR+1
1190     GOTO 1330
1200   IF NOT(TOKEN.CODE>=12 AND TOKEN.CODE<=15) THEN 1270
1210     CHKSW=0
1220     WHILE SPTR<>1 AND TOKEN.CODE¥2<=ASC(OPRSTK$(SPTR-1))¥2
1230       POL$(PPTR)=OPRSTK$(SPTR-1): PPTR=PPTR+1: SPTR=SPTR-1
1240     WEND
1250     OPRSTK$(SPTR)=CHR$(TOKEN.CODE)+TOKEN$: SPTR=SPTR+1
1260     GOTO 1330
1270   IF NOT(TOKEN$=")") THEN LFLG=0: GOTO 1330
1280     WHILE SPTR<>1 AND MID$(OPRSTK$(SPTR-1),2)<>"("
1290       POL$(PPTR)=OPRSTK$(SPTR-1): PPTR=PPTR+1: SPTR=SPTR-1
1300     WEND
1310     IF SPTR=1 THEN *SYNTAX.ERROR
1320     SPTR=SPTR-1
1330 WEND
1340 IF TOKEN.CODE<>0 THEN *SYNTAX.ERROR
1350 WHILE SPTR<>1
1360   IF MID$(OPRSTK$(SPTR-1),2)="(" THEN *SYNTAX.ERROR
1370   POL$(PPTR)=OPRSTK$(SPTR-1): PPTR=PPTR+1: SPTR=SPTR-1
1380 WEND
1390 PRINT "逆ポーランド記法 : ";
1400 FOR I=0 TO PPTR-1
1410   PRINT MID$(POL$(I),2);" ";
1420 NEXT
1430 PRINT
1440 '==========================================
1450 '    逆ポーランド記法の式よりオブジェクト・コードを生成
1460 '==========================================
1470 PRINT
1480 PRINT "-----<  逆ポーランド記法の式よりオブジェクト・コードを生成  >-----"
1490 PRINT
```

```
1500 PRINT "   **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **"
1510 PRINT
1520 SYMTBL$(0)="$"
1530 LOCADR=&H1000
1540 PSHOTF=0
1550 SPTR=0
1560 IF NOT(PPTR=1) THEN 1610
1570   X=ASC(POL$(0)): X$=MID$(POL$(0),2)
1580   IF X=1 THEN VARNA$=X$: GOSUB *LDHLVAR: GOTO 1970
1590   IF X=2 THEN CONS$=X$:  GOSUB *LDHLCNS: GOTO 1970
1600   GOTO *GEN.ERROR
1610 FOR I=0 TO PPTR-1
1620   TOKEN.CODE=ASC(POL$(I)): TOKEN$=MID$(POL$(I),2)
1630   IF NOT(TOKEN.CODE=1 OR TOKEN.CODE=2) THEN 1660
1640     OPRSTK$(SPTR)=POL$(I): SPTR=SPTR+1
1650     GOTO 1950
1660   X=ASC(OPRSTK$(SPTR-2)): X$=MID$(OPRSTK$(SPTR-2),2)
1670   Y=ASC(OPRSTK$(SPTR-1)): Y$=MID$(OPRSTK$(SPTR-1),2)
1680   IF NOT(TOKEN$="+") THEN 1740
1690     GOSUB *ADD.MUL
1700     PRINT FNH4$(LOCADR);" 19";              ' ADD  HL,DE
1710     PRINT TAB(15);"ADD     HL,DE"
1720     LOCADR=LOCADR+1
1730     GOTO 1930
1740   IF NOT(TOKEN$="-") THEN 1820
1750     GOSUB *SUB.DIV
1760     PRINT FNH4$(LOCADR);" B7";              ' OR   A
1770     PRINT TAB(15);"OR      A"
1780     PRINT FNH4$(LOCADR+1);" ED52";          ' SBC  HL,DE
1790     PRINT TAB(15);"SBC     HL,DE"
1800     LOCADR=LOCADR+3
1810     GOTO 1930
1820   IF NOT(TOKEN$="*") THEN 1880
1830     GOSUB *ADD.MUL
1840     PRINT FNH4$(LOCADR);" CD0030";          ' CALL IMUL
1850     PRINT TAB(15);"CALL    IMUL"
1860     LOCADR=LOCADR+3
1870     GOTO 1930
1880   IF NOT(TOKEN$="/") THEN *GEN.ERROR
1890     GOSUB *SUB.DIV
1900     PRINT FNH4$(LOCADR);" CD0330";          ' CALL IDIV
1910     PRINT TAB(15);"CALL    IDIV"
1920     LOCADR=LOCADR+3
1930   SPTR=SPTR-2
1940   OPRSTK$(SPTR)=CHR$(3)+" ": SPTR=SPTR+1
1950 NEXT I
1960 '
1970 PRINT
1980 PRINT "   **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **"
1990 PRINT
2000 FOR I=0 TO 19
2010   IF SYMTBL$(I)="$" THEN 2040
2020   PRINT FNH4$(&H4000+I*2);" ";SYMTBL$(I)
2030 NEXT I
2040 PRINT
2050 END
2060 '----------  加算および乗算  ----------
2070 *ADD.MUL
2080 IF NOT(X=1 AND Y=1) THEN 2130
2090   GOSUB *PUSHHL
2100   VARNA$=X$: GOSUB *LDHLVAR
2110   VARNA$=Y$: GOSUB *LDDEVAR
2120   RETURN
2130 IF NOT(X=1 AND Y=2) THEN 2180
2140   GOSUB *PUSHHL
2150   VARNA$=X$: GOSUB *LDHLVAR
2160   CONS$=Y$:  GOSUB *LDDECNS
2170   RETURN
```

```
2180 IF NOT(X=1 AND Y=3) THEN 2210
2190   VARNA$=X$: GOSUB *LDDEVAR
2200   RETURN
2210 IF NOT(X=2 AND Y=1) THEN 2260
2220   GOSUB *PUSHHL
2230   VARNA$=Y$: GOSUB *LDHLVAR
2240   CONS$=X$:  GOSUB *LDDECNS
2250   RETURN
2260 IF NOT(X=2 AND Y=2) THEN 2310
2270   GOSUB *PUSHHL
2280   CONS$=X$: GOSUB *LDHLCNS
2290   CONS$=Y$: GOSUB *LDDECNS
2300   RETURN
2310 IF NOT(X=2 AND Y=3) THEN 2340
2320   CONS$=X$: GOSUB *LDDECNS
2330   RETURN
2340 IF NOT(X=3 AND Y=1) THEN 2370
2350   VARNA$=Y$: GOSUB *LDDEVAR
2360   RETURN
2370 IF NOT(X=3 AND Y=2) THEN 2400
2380   CONS$=Y$: GOSUB *LDDECNS
2390   RETURN
2400 IF NOT(X=3 AND Y=3) THEN *GEN.ERROR
2410   GOSUB *POPDE
2420   RETURN
2430 '----------  減算および除算  ----------
2440 *SUB.DIV
2450 IF NOT(X=1 AND Y=1) THEN 2500
2460   GOSUB *PUSHHL
2470   VARNA$=X$: GOSUB *LDHLVAR
2480   VARNA$=Y$: GOSUB *LDDEVAR
2490   RETURN
2500 IF NOT(X=1 AND Y=2) THEN 2550
2510   GOSUB *PUSHHL
2520   VARNA$=X$: GOSUB *LDHLVAR
2530   CONS$=Y$:  GOSUB *LDDECNS
2540   RETURN
2550 IF NOT(X=1 AND Y=3) THEN 2590
2560   GOSUB *EXDEHL
2570   VARNA$=X$: GOSUB *LDHLVAR
2580   RETURN
2590 IF NOT(X=2 AND Y=1) THEN 2640
2600   GOSUB *PUSHHL
2610   CONS$=X$:  GOSUB *LDHLCNS
2620   VARNA$=Y$: GOSUB *LDDEVAR
2630   RETURN
2640 IF NOT(X=2 AND Y=2) THEN 2690
2650   GOSUB *PUSHHL
2660   CONS$=X$: GOSUB *LDHLCNS
2670   CONS$=Y$: GOSUB *LDDECNS
2680   RETURN
2690 IF NOT(X=2 AND Y=3) THEN 2730
2700   GOSUB *EXDEHL
2710   CONS$=X$: GOSUB *LDHLCNS
2720   RETURN
2730 IF NOT(X=3 AND Y=1) THEN 2760
2740   VARNA$=Y$: GOSUB *LDDEVAR
2750   RETURN
2760 IF NOT(X=3 AND Y=2) THEN 2790
2770   CONS$=Y$: GOSUB *LDDECNS
2780   RETURN
2790 IF NOT(X=3 AND Y=3) THEN *GEN.ERROR
2800   GOSUB *EXDEHL
2810   GOSUB *POPHL
2820   RETURN
2830 '---------------------------------------------
2840 *LDHLVAR              ' LD  HL,( 変数のアドレス )
2850  GOSUB *SYMTBL.SEA
```

```
2860  PRINT FNH4$(LOCADR);' 2A';FNRH4$(VARADR);
2870  PRINT TAB(15);'LD       HL,(';FNH4$(VARADR);'H)'
2880  LOCADR=LOCADR+3
2890  RETURN
2900 *LDDEVAR                ' LD  DE,( 変数のアドレス )
2910  GOSUB *SYMTBL.SEA
2920  PRINT FNH4$(LOCADR);' ED5B';FNRH4$(VARADR);
2930  PRINT TAB(15);'LD       DE,(';FNH4$(VARADR);'H)'
2940  LOCADR=LOCADR+4
2950  RETURN
2960 *LDHLCNS                ' LD  HL,定数
2970  CONS!=VAL(CONS$)
2980  IF CONS!>&H7FFF THEN *GEN.ERROR
2990  PRINT FNH4$(LOCADR);' 21';FNRH4$(CONS!);
3000  PRINT TAB(15);'LD       HL,';CONS$
3010  LOCADR=LOCADR+3
3020  RETURN
3030 *LDDECNS                ' LD  DE,定数
3040  CONS!=VAL(CONS$)
3050  IF CONS!>&H7FFF THEN *GEN.ERROR
3060  PRINT FNH4$(LOCADR);' 11';FNRH4$(CONS!);
3070  PRINT TAB(15);'LD       DE,';CONS$
3080  LOCADR=LOCADR+3
3090  RETURN
3100 *EXDEHL                 ' EX  DE,HL
3110  PRINT FNH4$(LOCADR);' EB';
3120  PRINT TAB(15);'EX       DE,HL'
3130  LOCADR=LOCADR+1
3140  RETURN
3150 *POPHL                  ' POP HL
3160  PRINT FNH4$(LOCADR);' E1';
3170  PRINT TAB(15);'POP      HL'
3180  LOCADR=LOCADR+1
3190  RETURN
3200 *POPDE                  ' POP DE
3210  PRINT FNH4$(LOCADR);' D1';
3220  PRINT TAB(15);'POP      DE'
3230  LOCADR=LOCADR+1
3240  RETURN
3250 *PUSHHL                 ' PUSH HL
3260  IF PSHOTF=0 THEN PSHOTF=1: RETURN
3270  PRINT FNH4$(LOCADR);' E5';
3280  PRINT TAB(15);'PUSH     HL'
3290  LOCADR=LOCADR+1
3300  RETURN
3310 '----------  記号表の検索および登録  ----------
3320 *SYMTBL.SEA
3330 FOR J=0 TO 19
3340   IF SYMTBL$(J)='$' THEN 3390
3350   IF SYMTBL$(J)=VARNA$ THEN 3410
3360 NEXT J
3370 PRINT: PRINT 'SYMBOL TABLE OVERFLOW'
3380 END
3390 SYMTBL$(J)=VARNA$
3400 SYMTBL$(J+1)='$'
3410 VARADR=&H4000+J*2
3420 RETURN
3430 '----------  構文 エラー  ----------
3440 *SYNTAX.ERROR
3450 PRINT: PRINT 'SYNTAX ERROR'
3460 END
3470 '----------  コード生成 エラー  ----------
3480 *GEN.ERROR
3490 PRINT: PRINT 'CODE GENERATION ERROR'
3500 END
5000 '==============================
5010 '      字句解析サブルーチン
5020 '==============================
```

```
5030 *GET.TOKEN
5040 '  -----  先行する空白を取る  -----
5050 WHILE CHAR$=" "
5060   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5070 WEND
5080 '  -----  行の終り  -----
5090 IF NOT(CHAR$=EOL$) THEN 5140
5100   TOKEN.CODE=0
5110   TOKEN$=""
5120   RETURN
5130 '  -----  変数名  -----
5140 IF NOT(CHAR$>="A" AND CHAR$<="Z") THEN 5230
5150   TOKEN.CODE=1
5160   TOKEN$=""
5170   WHILE (CHAR$>="A" AND CHAR$<="Z") OR (CHAR$>="0" AND CHAR$<="9")
5180     TOKEN$=TOKEN$+CHAR$
5190     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5200   WEND
5210   RETURN
5220 '  -----  数値定数  -----
5230 IF NOT(CHAR$>="0" AND CHAR$<="9") THEN 5320
5240   TOKEN.CODE=2
5250   TOKEN$=""
5260   WHILE CHAR$>="0" AND CHAR$<="9"
5270     TOKEN$=TOKEN$+CHAR$
5280     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5290   WEND
5300   RETURN
5310 '  -----  区切り記号  -----
5320 DELIM$="()+-*/"
5330 FOR I=1 TO LEN(DELIM$)
5340   IF CHAR$=MID$(DELIM$,I,1) THEN 5370
5350 NEXT I
5360 GOTO *TOKEN.ERROR
5370 TOKEN.CODE=9+I
5380 TOKEN$=CHAR$
5390 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5400 RETURN
5410 '  -----  エラー  -----
5420 *TOKEN.ERROR
5430     PRINT LEFT$(SOU.LINE$,LEN(SOU.LINE$)-1)
5440     PRINT TAB(CP-2);"^ Error"
5450     END
6000 '========================================
6010 '         ソース・プログラム(式)の入力
6020 '========================================
6030 *READ.SOU
6040 LINE INPUT "式を入力 : ",IN.EXPR$
6050 SOU.LINE$=IN.EXPR$+EOL$
6060 CHAR$=MID$(SOU.LINE$,1,1): CP=2
6070 RETURN
```

●実行例3-3

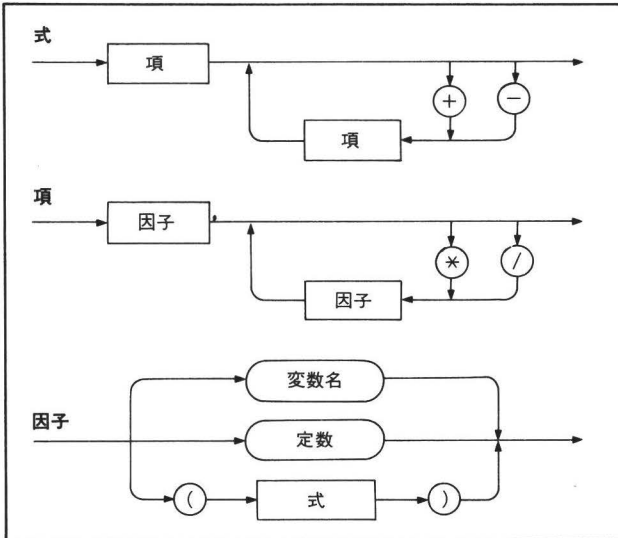
```
式を入力 : A + B/( C-2 ) + D*E

-----<  普通の式を逆ポーランド記法の式に変換  >-----

入力された式     : A + B/( C-2 ) + D*E
逆ポーランド記法 : A B C 2 - / + D E * +

-----<  逆ポーランド記法の式よりオブジェクト・コードを生成  >-----

  **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **

1000 2A0040     LD      HL,(4000H)
1003 110200     LD      DE,2
1006 B7         OR      A
1007 ED52       SBC     HL,DE
1009 EB         EX      DE,HL
100A 2A0240     LD      HL,(4002H)
100D CD0330     CALL    IDIV
1010 ED5B0440   LD      DE,(4004H)
1014 19         ADD     HL,DE
1015 E5         PUSH    HL
1016 2A0640     LD      HL,(4006H)
1019 ED5B0840   LD      DE,(4008H)
101D CD0030     CALL    IMUL
1020 D1         POP     DE
1021 19         ADD     HL,DE

  **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **

4000 C
4002 B
4004 A
4006 D
4008 E
```

3-3-3 再帰的な方法で式をコンパイルする

これは構文図で書かれたような再帰的な構文をそのままプログラムにする方法です。構文図からプログラムをつくる場合の規則については，**3－2－2**の後半部を見てください。本書第2部のコンパイラはこの方法でプログラミングされています。

この方法では，字句解析部から構文解析部へ渡されるトークンを終端記号とした新たな構文図をつくります。そして，この構文図から**3－2－2**で説明した変換規則をもとにして流れ図やプログラムをつくります。基本的には，この作業でコンパイラの構文解析部をつくることができます。

たとえば，**3－3－2**の60ページのような式の構文があ

ったとします。こういった構文を持つ式で，字句解析からトークンとして構文解析へ渡されるのは変数名，定数，および空白を除いた区切り記号です。すると，構文解析で処理しなければならない構文は加減演算子，乗除演算子，因子，項，式となります。トークンに相当するもの（変数名，定数，空白を除いた区切り記号）を終端記号としてこれを構文図に書くと，

となります。あとはこの構文図から流れ図やプログラムをつくればよいのです。この構文図を流れ図にすると**図3-11**のようになります。

この流れ図をプログラムにしたものが**リスト3-4**です。このプログラムでは，変数や演算といったものをすべて16ビットの符号つき整数として扱い，演算時のオーバーフローは考えていません。**図3-11**の※１～６に相当する部分では次のようなコードを生成するようにしました。このオブジェクト・コードでは，中間結果を一時スタックに記憶するようにしています。

※1　変数をロードするオブジェクト・コードを生成

```
PUSH HL
LD    HL,（変数のアドレス）
```

ただし“PUSH HL”は初めてのオブジェクト・コードの生成のときには出力しない。

※2　定数をロードするオブジェクト・コードを生成

```
PUSH HL
LD    HL, 定数
```

ただし“PUSH HL”は初めてのオブジェクト・コードの生成のときには出力しない。

※3　加算のオブジェクト・コードを生成

```
POP DE
ADD HL, DE
```

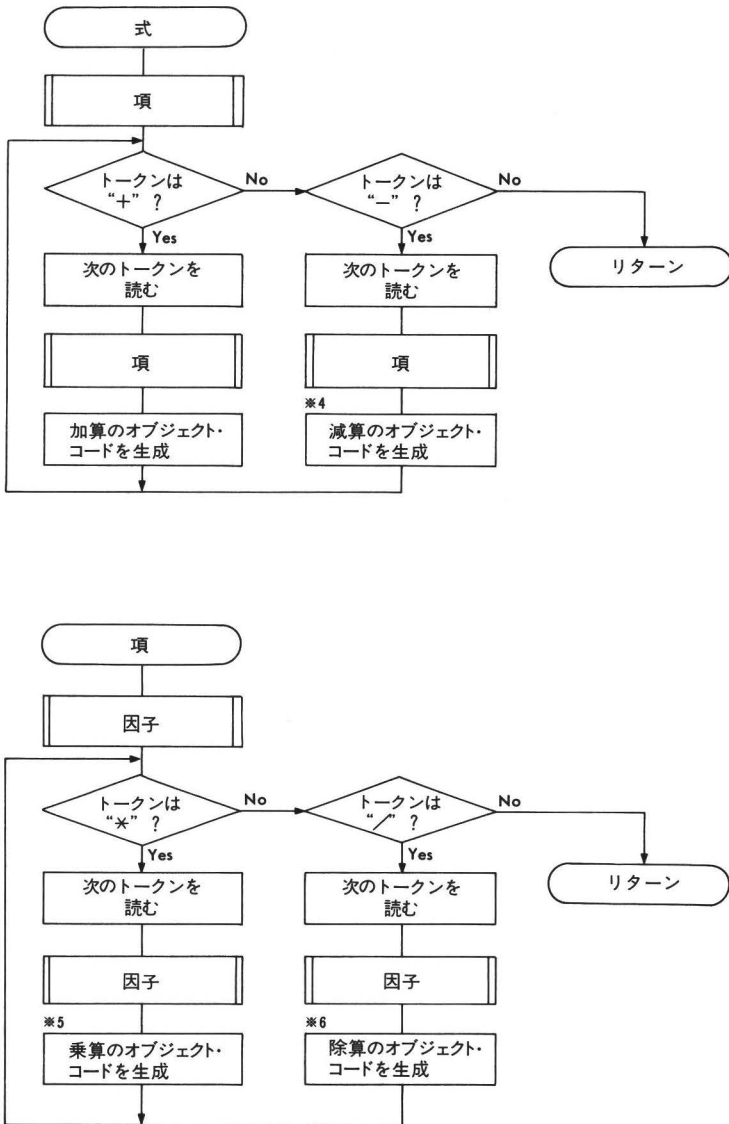
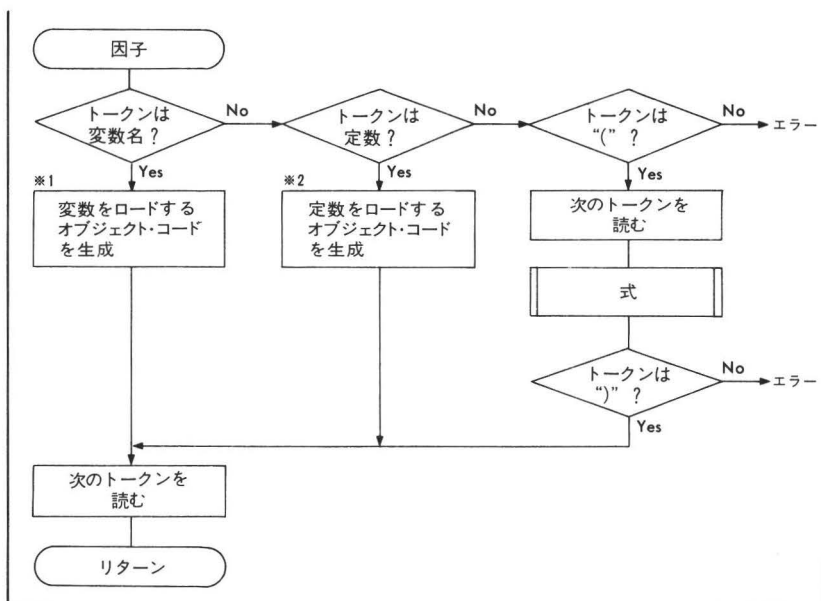
※4　減算のオブジェクト・コードを生成

```
EX  DE, HL
POP HL
OR  A
SBC HL, DE
```

※5　乗算のオブジェクト・コードを生成

```
POP  DE
CALL IMUL
```

IMULは乗算 (HL ← HL * DE) を行うランタイム・ルーチン

図3-11 式を再帰的にコンパイルする場合の流れ図

※6　除算のオブジェクト・コードを生成

```
EX DE,HL
POP HL
CALL IDIV
```

IDIVは除算（HL←HL／DE）を行うランタイム・ルーチン

オブジェクト・プログラムは16進数で1000番地から，変数は4000番地から始まるようになっています。そして，ランタイム・ルーチンとして3000番地に乗算（HL←HL＊DE），3003番地に除算（HL←HL／DE）ルーチンがあるものとします。

●リスト3-4

```
100 '****************************************
110 '
120 '    プログラムを再帰的に作ることで式をコンパイルする
130 '
140 '            （Z80　CPUの機械語に変換）
150 '
160 '****************************************
170 '
180 DEFINT A-Z
190 DIM SYMTBL$(20)
200 DEF FNH2$(X!)=RIGHT$("0"+HEX$(X!),2)
210 DEF FNH4$(X!)=RIGHT$("000"+HEX$(X!),4)
220 DEF FNRH4$(X!)=RIGHT$(FNH4$(X!),2)+LEFT$(FNH4$(X!),2)
230 '
240 EOL$=CHR$(0)
250 GOSUB *READ.SOU
260 '
270 PRINT
280 PRINT "入力された式  : ";IN.EXPR$
290 PRINT
300 SYMTBL$(0)="$"
310 LOCADR=&H1000
320 PSHOTF=0
330 '
340 PRINT "   **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **"
350 PRINT
360 GOSUB *GET.TOKEN
370 GOSUB *EXPRE
380 IF NOT(TOKEN.CODE=0) THEN *SYNTAX.ERROR
390 '
400 PRINT
410 PRINT "   **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **"
420 PRINT
430 FOR I=0 TO 19
440   IF SYMTBL$(I)="$" THEN 470
450   PRINT FNH4$(&H4000+I*2);" ";SYMTBL$(I)
460 NEXT I
470 PRINT
480 END
1000 '========================================
1010 '                式の構文を解析
1020 '========================================
1030 '
1040 '----------  式  ----------
1050 *EXPRE
1060  GOSUB *TERM.
1070  IF NOT(TOKEN$="+") THEN 1120
1080    GOSUB *GET.TOKEN
1090    GOSUB *TERM.
1100    GOSUB *ADDOBJ
1110    GOTO 1070
1120  IF NOT(TOKEN$="-") THEN RETURN
1130    GOSUB *GET.TOKEN
1140    GOSUB *TERM.
1150    GOSUB *SUBOBJ
1160    GOTO 1070
1170 '----------  項  ----------
1180 *TERM.
1190  GOSUB *FACTOR
1200  IF NOT(TOKEN$="*") THEN 1250
1210    GOSUB *GET.TOKEN
1220    GOSUB *FACTOR
1230    GOSUB *MULOBJ
1240    GOTO 1200
1250  IF NOT(TOKEN$="/") THEN RETURN
1260    GOSUB *GET.TOKEN
1270    GOSUB *FACTOR
```

```
1280    GOSUB *DIVOBJ
1290    GOTO 1200
1300 '----------  因子  ----------
1310 *FACTOR
1320  IF TOKEN.CODE=1 THEN GOSUB *LDHLVAR: GOTO 1380
1330  IF TOKEN.CODE=2 THEN GOSUB *LDHLCNS: GOTO 1380
1340  IF NOT(TOKEN$="(") THEN *SYNTAX.ERROR
1350    GOSUB *GET.TOKEN
1360    GOSUB *EXPRE
1370    IF NOT(TOKEN$=")") THEN *SYNTAX.ERROR
1380  GOSUB *GET.TOKEN
1390  RETURN
2000 '========================================
2010 '         オブジェクト・コードを生成
2020 '========================================
2030 '
2040 *LDHLVAR  '----------
2050  GOSUB *SYMTBL.SEA
2060  GOSUB *PUSHOUT                          ' PUSH HL
2070  PRINT FNH4$(LOCADR);" 2A";FNRH4$(VARADR); ' LD HL,( 変数のアドレス )
2080  PRINT TAB(15);"LD      HL,(";FNH4$(VARADR);"H)"
2090  LOCADR=LOCADR+3
2100  RETURN
2110 *LDHLCNS  '----------
2120  CONS!=VAL(TOKEN$)
2130  IF CONS!>&H7FFF THEN *GEN.ERROR
2140  GOSUB *PUSHOUT                          ' PUSH HL
2150  PRINT FNH4$(LOCADR);" 21";FNRH4$(CONS!);  ' LD HL,定数
2160  PRINT TAB(15);"LD      HL,";TOKEN$
2170  LOCADR=LOCADR+3
2180  RETURN
2190 *ADDOBJ  '----------
2200  PRINT FNH4$(LOCADR);" D1";               ' POP DE
2210  PRINT TAB(15);"POP     DE"
2220  PRINT FNH4$(LOCADR+1);" 19";             ' ADD HL,DE
2230  PRINT TAB(15);"ADD     HL,DE"
2240  LOCADR=LOCADR+2
2250  RETURN
2260 *SUBOBJ  '----------
2270  PRINT FNH4$(LOCADR);" EB";               ' EX DE,HL
2280  PRINT TAB(15);"EX      DE,HL"
2290  PRINT FNH4$(LOCADR+1);" E1";             ' POP HL
2300  PRINT TAB(15);"POP     HL"
2310  PRINT FNH4$(LOCADR+2);" B7";             ' OR A
2320  PRINT TAB(15);"OR      A"
2330  PRINT FNH4$(LOCADR+3);" ED52";           ' SBC HL,DE
2340  PRINT TAB(15);"SBC     HL,DE"
2350  LOCADR=LOCADR+5
2360  RETURN
2370 *MULOBJ  '----------
2380  PRINT FNH4$(LOCADR);" D1";               ' POP DE
2390  PRINT TAB(15);"POP     DE"
2400  PRINT FNH4$(LOCADR+1);" CD0030";       ' ' CALL IMUL
2410  PRINT TAB(15);"CALL    IMUL"
2420  LOCADR=LOCADR+4
2430  RETURN
2440 *DIVOBJ  '----------
2450  PRINT FNH4$(LOCADR);" EB";               ' EX DE,HL
2460  PRINT TAB(15);"EX      DE,HL"
2470  PRINT FNH4$(LOCADR+1);" E1";             ' POP HL
2480  PRINT TAB(15);"POP     HL"
2490  PRINT FNH4$(LOCADR+2);" CD0330";         ' CALL IDIV
2500  PRINT TAB(15);"CALL    IDIV"
2510  LOCADR=LOCADR+5
2520  RETURN
2530 '
2540 '----------  " PUSH HL " の出力  ----------
2550 *PUSHOUT
```

```
2560 IF PSHOTF=0 THEN PSHOTF=1: RETURN
2570 PRINT FNH4$(LOCADR);" E5";                    ' PUSH HL
2580 PRINT TAB(15);"PUSH    HL"
2590 LOCADR=LOCADR+1
2600 RETURN
3000 '----------  記号表の検索および登録  ----------
3010 *SYMTBL.SEA
3020 FOR J=0 TO 19
3030   IF SYMTBL$(J)="$" THEN 3080
3040   IF SYMTBL$(J)=TOKEN$ THEN 3100
3050 NEXT J
3060 PRINT: PRINT "SYMBOL TABLE OVERFLOW"
3070 END
3080 SYMTBL$(J)=TOKEN$
3090 SYMTBL$(J+1)="$"
3100 VARADR=&H4000+J*2
3110 RETURN
3120 '----------  構文 エラー  ----------
3130 *SYNTAX.ERROR
3140 PRINT: PRINT "SYNTAX ERROR"
3150 END
3160 '----------  コード生成 エラー  ----------
3170 *GEN.ERROR
3180 PRINT: PRINT "CODE GENERATION ERROR"
3190 END
4000 '==============================
4010 '       字句解析サブルーチン
4020 '==============================
4030 *GET.TOKEN
4040 '  -----  先行する空白を取る  -----
4050 WHILE CHAR$=" "
4060   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
4070 WEND
4080 '  -----  行の終り  -----
4090 IF NOT(CHAR$=EOL$) THEN 4140
4100   TOKEN.CODE=0
4110   TOKEN$=""
4120   RETURN
4130 '  -----  変数名  -----
4140 IF NOT(CHAR$>="A" AND CHAR$<="Z") THEN 4230
4150   TOKEN.CODE=1
4160   TOKEN$=""
4170   WHILE (CHAR$>="A" AND CHAR$<="Z") OR (CHAR$>="0" AND CHAR$<="9")
4180     TOKEN$=TOKEN$+CHAR$
4190     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
4200   WEND
4210   RETURN
4220 '  -----  数値定数  -----
4230 IF NOT(CHAR$>="0" AND CHAR$<="9") THEN 4320
4240   TOKEN.CODE=2
4250   TOKEN$=""
4260   WHILE CHAR$>="0" AND CHAR$<="9"
4270     TOKEN$=TOKEN$+CHAR$
4280     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
4290   WEND
4300   RETURN
4310 '  -----  区切り記号  -----
4320 DELIM$="()+-*/"
4330 FOR I=1 TO LEN(DELIM$)
4340   IF CHAR$=MID$(DELIM$,I,1) THEN 4370
4350 NEXT I
4360 GOTO *TOKEN.ERROR
4370 TOKEN.CODE=9+I
4380 TOKEN$=CHAR$
4390 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
4400 RETURN
4410 '  -----  エラー  -----
4420 *TOKEN.ERROR
```

```
4430      PRINT LEFT$(SOU.LINE$,LEN(SOU.LINE$)-1)
4440      PRINT TAB(CP-2);"^ Error"
4450      END
5000 '========================================
5010 '        ソース・プログラム（式）の入力
5020 '========================================
5030 *READ.SOU
5040 LINE INPUT "式を入力 : ",IN.EXPR$
5050 SOU.LINE$=IN.EXPR$+EOL$
5060 CHAR$=MID$(SOU.LINE$,1,1): CP=2
5070 RETURN
```

●実行例3-4

```
式を入力 : A + B/( C-2 ) + D*E

入力された式  : A + B/( C-2 ) + D*E

  **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **

1000 2A0040     LD      HL,(4000H)
1003 E5         PUSH    HL
1004 2A0240     LD      HL,(4002H)
1007 E5         PUSH    HL
1008 2A0440     LD      HL,(4004H)
100B E5         PUSH    HL
100C 210200     LD      HL,2
100F EB         EX      DE,HL
1010 E1         POP     HL
1011 B7         OR      A
1012 ED52       SBC     HL,DE
1014 EB         EX      DE,HL
1015 E1         POP     HL
1016 CD0330     CALL    IDIV
1019 D1         POP     DE
101A 19         ADD     HL,DE
101B E5         PUSH    HL
101C 2A0640     LD      HL,(4006H)
101F E5         PUSH    HL
1020 2A0840     LD      HL,(4008H)
1023 D1         POP     DE
1024 CD0030     CALL    IMUL
1027 D1         POP     DE
1028 19         ADD     HL,DE

  **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **

4000 A
4002 B
4004 C
4006 D
4008 E
```

実行例3-4がプログラムの実行結果です。このオブジェクト・プログラムは逆ポーランド記法を用いた**実行例3-2**(70ページ) と同じになっています。これは，オブジェクト・コードの生成段階で，変数のロード，定数のロード，加算，減算，乗算，除算といったオブジェクト・コードを，逆ポーランド記法を用いた式のコンパイルのプ

ログラムと同一にしたためです。この場合，コード生成部は，逆ポーランド記法の式を一括してオブジェクト・コードに変換するのか，サブルーチンとして構文解析から呼び出されるのかが異なるだけです。**リスト3-2**と**リスト3-4**のコード生成部を見比べればわかると思います。

そのためここでも，無駄な PUSH，POP 命令が生成されています。そこで，**3-3-2**で使った最適化をします。つまり，**図3-11**の※1～6に相当する部分で次のような操作を行います。

※1および※2　変数名や定数はスタックへ PUSH する。

※3～※6　スタックから二つの変数名や定数，あるいは中間結果を示すマークをＰＯＰし，**表3-2**(73ページ)を用いてオブジェクト・コードを生成する。この表の使いかたは**3-3-2**のときと同じである。

以上のことを取り入れてプログラムにしたのが**リスト3-5**です。そして，**実行例3-5**がこのプログラムの実行結果です。このプログラムの場合，同じ**表3-2**を使ったため，**実行例3-3** (80ページ) のオブジェクト・プログラムと同じものが生成されています。

●リスト3-5

```
100 '****************************************
110 '
120 '    プログラムを再帰的に作ることで式をコンパイルする
130 '                簡単な最適化を行なった例
140 '
150 '           (Z80  CPUの機械語に変換)
160 '
170 '****************************************
180 '
190 DEFINT A-Z
200 DIM OPRSTK$(50),SYMTBL$(20)
210 DEF FNH2$(X!)=RIGHT$("0"+HEX$(X!),2)
220 DEF FNH4$(X!)=RIGHT$("000"+HEX$(X!),4)
230 DEF FNRH4$(X!)=RIGHT$(FNH4$(X!),2)+LEFT$(FNH4$(X!),2)
240 '
250 EOL$=CHR$(0)
260 GOSUB *READ.SOU
270 '
280 PRINT
```

```
290 PRINT "入力された式  :  ";IN.EXPR$
300 PRINT
310 SYMTBL$(0)="$"
320 LOCADR=&H1000
330 PSHOTF=0: SPTR=0
340 '
350 PRINT "  **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **"
360 PRINT
370 GOSUB *GET.TOKEN
380 GOSUB *EXPRE
390 IF NOT(TOKEN.CODE=0) THEN *SYNTAX.ERROR
400 IF ASC(OPRSTK$(0))=3 THEN 440
410   X=ASC(OPRSTK$(0)): X$=MID$(OPRSTK$(0),2)
420   IF X=1 THEN VARNA$=X$: GOSUB *LDHLVAR: GOTO 440
430   CONS$=X$: GOSUB *LDHLCNS
440 '
450 PRINT
460 PRINT "  **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **"
470 PRINT
480 FOR I=0 TO 19
490   IF SYMTBL$(I)="$" THEN 520
500   PRINT FNH4$(&H4000+I*2);"  ";SYMTBL$(I)
510 NEXT I
520 PRINT
530 END
1000 '========================================
1010 '              式の構文を解析
1020 '========================================
1030 '
1040 '----------  式  ----------
1050 *EXPRE
1060  GOSUB *TERM.
1070  IF NOT(TOKEN$="+") THEN 1120
1080    GOSUB *GET.TOKEN
1090    GOSUB *TERM.
1100    GOSUB *ADDOBJ
1110    GOTO 1070
1120  IF NOT(TOKEN$="-") THEN RETURN
1130    GOSUB *GET.TOKEN
1140    GOSUB *TERM.
1150    GOSUB *SUBOBJ
1160    GOTO 1070
1170 '----------  項  ----------
1180 *TERM.
1190  GOSUB *FACTOR
1200  IF NOT(TOKEN$="*") THEN 1250
1210    GOSUB *GET.TOKEN
1220    GOSUB *FACTOR
1230    GOSUB *MULOBJ
1240    GOTO 1200
1250  IF NOT(TOKEN$="/") THEN RETURN
1260    GOSUB *GET.TOKEN
1270    GOSUB *FACTOR
1280    GOSUB *DIVOBJ
1290    GOTO 1200
1300 '----------  因子  ----------
1310 *FACTOR
1320  IF TOKEN.CODE=1 OR TOKEN.CODE=2 THEN GOSUB *STKPUSH: GOTO
1330  IF NOT(TOKEN$="(") THEN *SYNTAX.ERROR
1340    GOSUB *GET.TOKEN
1350    GOSUB *EXPRE
1360    IF NOT(TOKEN$=")") THEN *SYNTAX.ERROR
1370  GOSUB *GET.TOKEN
1380  RETURN
1390 '==========================================
1400 '        オブジェクト・コードを生成
1410 '==========================================
1420 '
```

```
1430 '
1440 *ADDOBJ
1450     GOSUB *ADD.MUL
1460     PRINT FNH4$(LOCADR);" 19";                    ' ADD  HL,DE
1470     PRINT TAB(15);"ADD     HL,DE"
1480     LOCADR=LOCADR+1
1490     RETURN
1500 *SUBOBJ
1510     GOSUB *SUB.DIV
1520     PRINT FNH4$(LOCADR);" B7";                    ' OR   A
1530     PRINT TAB(15);"OR      A"
1540     PRINT FNH4$(LOCADR+1);" ED52";                ' SBC  HL,DE
1550     PRINT TAB(15);"SBC     HL,DE"
1560     LOCADR=LOCADR+3
1570     RETURN
1580 *MULOBJ
1590     GOSUB *ADD.MUL
1600     PRINT FNH4$(LOCADR);" CD0030";                ' CALL IMUL
1610     PRINT TAB(15);"CALL    IMUL"
1620     LOCADR=LOCADR+3
1630     RETURN
1640 *DIVOBJ
1650     GOSUB *SUB.DIV
1660     PRINT FNH4$(LOCADR);" CD0330";                ' CALL IDIV
1670     PRINT TAB(15);"CALL    IDIV"
1680     LOCADR=LOCADR+3
1690     RETURN
1700 '
1710 '----------  加算および乗算  ----------
1720 *ADD.MUL
1730 GOSUB *POPSTK
1740 IF NOT(X=1 AND Y=1) THEN 1790
1750   GOSUB *PUSHHL
1760   VARNA$=X$: GOSUB *LDHLVAR
1770   VARNA$=Y$: GOSUB *LDDEVAR
1780   RETURN
1790 IF NOT(X=1 AND Y=2) THEN 1840
1800   GOSUB *PUSHHL
1810   VARNA$=X$: GOSUB *LDHLVAR
1820   CONS$=Y$:  GOSUB *LDDECNS
1830   RETURN
1840 IF NOT(X=1 AND Y=3) THEN 1870
1850   VARNA$=X$: GOSUB *LDDEVAR
1860   RETURN
1870 IF NOT(X=2 AND Y=1) THEN 1920
1880   GOSUB *PUSHHL
1890   VARNA$=Y$: GOSUB *LDHLVAR
1900   CONS$=X$:  GOSUB *LDDECNS
1910   RETURN
1920 IF NOT(X=2 AND Y=2) THEN 1970
1930   GOSUB *PUSHHL
1940   CONS$=X$: GOSUB *LDHLCNS
1950   CONS$=Y$: GOSUB *LDDECNS
1960   RETURN
1970 IF NOT(X=2 AND Y=3) THEN 2000
1980   CONS$=X$: GOSUB *LDDECNS
1990   RETURN
2000 IF NOT(X=3 AND Y=1) THEN 2030
2010   VARNA$=Y$: GOSUB *LDDEVAR
2020   RETURN
2030 IF NOT(X=3 AND Y=2) THEN 2060
2040   CONS$=Y$: GOSUB *LDDECNS
2050   RETURN
2060 IF NOT(X=3 AND Y=3) THEN *GEN.ERROR
2070   GOSUB *POPDE
2080   RETURN
2090 '----------  減算および除算  ----------
2100 *SUB.DIV
```

```
2110 GOSUB *POPSTK
2120 IF NOT(X=1 AND Y=1) THEN 2170
2130   GOSUB *PUSHHL
2140   VARNA$=X$: GOSUB *LDHLVAR
2150   VARNA$=Y$: GOSUB *LDDEVAR
2160   RETURN
2170 IF NOT(X=1 AND Y=2) THEN 2220
2180   GOSUB *PUSHHL
2190   VARNA$=X$: GOSUB *LDHLVAR
2200   CONS$=Y$:  GOSUB *LDDECNS
2210   RETURN
2220 IF NOT(X=1 AND Y=3) THEN 2260
2230   GOSUB *EXDEHL
2240   VARNA$=X$: GOSUB *LDHLVAR
2250   RETURN
2260 IF NOT(X=2 AND Y=1) THEN 2310
2270   GOSUB *PUSHHL
2280   CONS$=X$:  GOSUB *LDHLCNS
2290   VARNA$=Y$: GOSUB *LDDEVAR
2300   RETURN
2310 IF NOT(X=2 AND Y=2) THEN 2360
2320   GOSUB *PUSHHL
2330   CONS$=X$: GOSUB *LDHLCNS
2340   CONS$=Y$: GOSUB *LDDECNS
2350   RETURN
2360 IF NOT(X=2 AND Y=3) THEN 2400
2370   GOSUB *EXDEHL
2380   CONS$=X$: GOSUB *LDHLCNS
2390   RETURN
2400 IF NOT(X=3 AND Y=1) THEN 2430
2410   VARNA$=Y$: GOSUB *LDDEVAR
2420   RETURN
2430 IF NOT(X=3 AND Y=2) THEN 2460
2440   CONS$=Y$: GOSUB *LDDECNS
2450   RETURN
2460 IF NOT(X=3 AND Y=3) THEN *GEN.ERROR
2470   GOSUB *EXDEHL
2480   GOSUB *POPHL
2490   RETURN
2500 '---------------------------------------------
2510 *LDHLVAR                ' LD  HL,( 変数のアドレス )
2520  GOSUB *SYMTBL.SEA
2530  PRINT FNH4$(LOCADR);" 2A";FNRH4$(VARADR);
2540  PRINT TAB(15);"LD      HL,(";FNH4$(VARADR);"H)"
2550  LOCADR=LOCADR+3
2560  RETURN
2570 *LDDEVAR                ' LD  DE,( 変数のアドレス )
2580  GOSUB *SYMTBL.SEA
2590  PRINT FNH4$(LOCADR);" ED5B";FNRH4$(VARADR);
2600  PRINT TAB(15);"LD      DE,(";FNH4$(VARADR);"H)"
2610  LOCADR=LOCADR+4
2620  RETURN
2630 *LDHLCNS                ' LD  HL,定数
2640  CONS!=VAL(CONS$)
2650  IF CONS!>&H7FFF THEN *GEN.ERROR
2660  PRINT FNH4$(LOCADR);" 21";FNRH4$(CONS!);
2670  PRINT TAB(15);"LD      HL,";CONS$
2680  LOCADR=LOCADR+3
2690  RETURN
2700 *LDDECNS                ' LD  DE,定数
2710  CONS!=VAL(CONS$)
2720  IF CONS!>&H7FFF THEN *GEN.ERROR
2730  PRINT FNH4$(LOCADR);" 11";FNRH4$(CONS!);
2740  PRINT TAB(15);"LD      DE,";CONS$
2750  LOCADR=LOCADR+3
2760  RETURN
2770 *EXDEHL                 ' EX  DE,HL
2780  PRINT FNH4$(LOCADR);" EB";
```

```
2790  PRINT TAB(15);'EX       DE,HL'
2800  LOCADR=LOCADR+1
2810  RETURN
2820 *POPHL                  ' POP HL
2830  PRINT FNH4$(LOCADR);' E1';
2840  PRINT TAB(15);'POP      HL'
2850  LOCADR=LOCADR+1
2860  RETURN
2870 *POPDE                  ' POP DE
2880  PRINT FNH4$(LOCADR);' D1';
2890  PRINT TAB(15);'POP      DE'
2900  LOCADR=LOCADR+1
2910  RETURN
2920 *PUSHHL                 ' PUSH HL
2930  IF PSHOTF=0 THEN PSHOTF=1: RETURN
2940  PRINT FNH4$(LOCADR);' E5';
2950  PRINT TAB(15);'PUSH     HL'
2960  LOCADR=LOCADR+1
2970  RETURN
4000 '
4010 '----------  変数および定数をスタックへPUSH  ----------
4020 *STKPUSH
4030  OPRSTK$(SPTR)=CHR$(TOKEN.CODE)+TOKEN$
4040  SPTR=SPTR+1
4050  RETURN
4060 '----------  変数および定数をスタックよりPOP  ----------
4070 *POPSTK
4080  X=ASC(OPRSTK$(SPTR-2)): X$=MID$(OPRSTK$(SPTR-2),2)
4090  Y=ASC(OPRSTK$(SPTR-1)): Y$=MID$(OPRSTK$(SPTR-1),2)
4100  OPRSTK$(SPTR-2)=CHR$(3)+' '
4110  SPTR=SPTR-1
4120  RETURN
4130 '----------  記号表の検索および登録  ----------
4140 *SYMTBL.SEA
4150 FOR J=0 TO 19
4160   IF SYMTBL$(J)='$' THEN 4210
4170   IF SYMTBL$(J)=VARNA$ THEN 4230
4180 NEXT J
4190 PRINT: PRINT 'SYMBOL TABLE OVERFLOW'
4200 END
4210 SYMTBL$(J)=VARNA$
4220 SYMTBL$(J+1)='$'
4230 VARADR=&H4000+J*2
4240 RETURN
4250 '----------  構文 エラー  ----------
4260 *SYNTAX.ERROR
4270 PRINT: PRINT 'SYNTAX ERROR'
4280 END
4290 '----------  コード生成 エラー  ----------
4300 *GEN.ERROR
4310 PRINT: PRINT 'CODE GENERATION ERROR'
4320 END
5000 '==============================
5010 '      字句解析サブルーチン
5020 '==============================
5030 *GET.TOKEN
5040 '  -----  先行する空白を取る  -----
5050 WHILE CHAR$=' '
5060   CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5070 WEND
5080 '  -----  行の終り  -----
5090 IF NOT(CHAR$=EOL$) THEN 5140
5100   TOKEN.CODE=0
5110   TOKEN$=''
5120   RETURN
5130 '  -----  変数名  -----
5140 IF NOT(CHAR$>='A' AND CHAR$<='Z') THEN 5230
5150   TOKEN.CODE=1
```

```
5160   TOKEN$=""
5170   WHILE (CHAR$>="A" AND CHAR$<="Z") OR (CHAR$>="0" AND CHAR$<="9")
5180     TOKEN$=TOKEN$+CHAR$
5190     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5200   WEND
5210   RETURN
5220 '  -----  数値定数  -----
5230 IF NOT(CHAR$>="0" AND CHAR$<="9") THEN 5320
5240   TOKEN.CODE=2
5250   TOKEN$=""
5260   WHILE CHAR$>="0" AND CHAR$<="9"
5270     TOKEN$=TOKEN$+CHAR$
5280     CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5290   WEND
5300   RETURN
5310 '  -----  区切り記号  -----
5320 DELIM$="()+-*/"
5330 FOR I=1 TO LEN(DELIM$)
5340   IF CHAR$=MID$(DELIM$,I,1) THEN 5370
5350 NEXT I
5360 GOTO *TOKEN.ERROR
5370 TOKEN.CODE=9+I
5380 TOKEN$=CHAR$
5390 CHAR$=MID$(SOU.LINE$,CP,1): CP=CP+1
5400 RETURN
5410 '  -----  エラー  -----
5420 *TOKEN.ERROR
5430     PRINT LEFT$(SOU.LINE$,LEN(SOU.LINE$)-1)
5440     PRINT TAB(CP-2);"^ Error"
5450     END
6000 '========================================
6010 '         ソース・プログラム（式）の入力
6020 '========================================
6030 *READ.SOU
6040 LINE INPUT "式を入力 : ",IN.EXPR$
6050 SOU.LINE$=IN.EXPR$+EOL$
6060 CHAR$=MID$(SOU.LINE$,1,1): CP=2
6070 RETURN
```

このように，構文の上から下へ解析していく方法，この例では〈式〉→〈項〉→〈因子〉→〈トークン〉へと解析していく方法のことをトップ・ダウン法（top down method）と呼びます。また，逆ポーランド記法のようにトークンの集まりからその構文を解析する方法をボトム・アップ法（bottom up method）と呼びます。

このうち割合簡単で失敗の少ないのは，トップ・ダウン法でしょう。なぜなら，構文図からそのままプログラムがつくれ，次にどのようなトークンがくるかがプログラムの流れから決まるために構文エラーが発見しやすい，すなわちエラーチェックのルーチンがつくりやすいからです。ボトム・アップ法では，ソース・プログラムから中間形にするのは簡単な処理で済むのですが，構文エラ

●実行例3-5

```
式を入力 : A + B/( C-2 ) + D*E

入力された式  : A + B/( C-2 ) + D*E

  **  ｵﾌﾞｼﾞｪｸﾄ･ﾌﾟﾛｸﾞﾗﾑ  **

1000 2A0040     LD      HL,(4000H)
1003 110200     LD      DE,2
1006 B7         OR      A
1007 ED52       SBC     HL,DE
1009 EB         EX      DE,HL
100A 2A0240     LD      HL,(4002H)
100D CD0330     CALL    IDIV
1010 ED5B0440   LD      DE,(4004H)
1014 19         ADD     HL,DE
1015 E5         PUSH    HL
1016 2A0640     LD      HL,(4006H)
1019 ED5B0840   LD      DE,(4008H)
101D CD0030     CALL    IMUL
1020 D1         POP     DE
1021 19         ADD     HL,DE

  **  ｼﾝﾎﾞﾙ･ﾃｰﾌﾞﾙ  **

4000 C
4002 B
4004 A
4006 D
4008 E
```

ーのチェックが，構文が複雑になるにつれて大変になってきます。それは，次にくるトークンを前のトークンから推測しなければならないからです。構文が複雑になればトークンの組み合わせも多くなります。

パソコンのコンパイラは普通アセンブラ言語でつくります。しかし，コンパイラを高水準言語でつくろうとすると，言語によっては再帰的なプログラムが文法的に許されない場合（FORTRANなど）があるという問題にぶつかります。そのようなときは無理してトップ・ダウン法を使うより，ボトム・アップ法のほうがコンパイラが小さく見やすくなります。

また，これまでの説明は式の中でも二項演算，それも四則演算についてでしたが，演算にはまだ関係演算，論理演算といったものがあります。これらの演算も二項演算については，これまで説明した四則演算と同様の方法でコンパイルすることができます。

3-3-4 代入文と単項演算子，配列，関数

ここでは変数への代入や単項演算子，配列，関数の呼び出しなどの処理について説明します。

〔1〕代入文

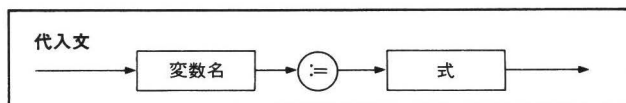
代入文は式の演算結果を変数に代入（ストア）するための文です。代入文は，FORTRANやBASICでは，

変数名 ＝ 式

と書き，Pascal や ALGOL では

変数名 :＝ 式

と書きます。代入される変数は“＝”または“：＝”の左辺に，式は右辺に書かれます。“＝”や“：＝”は右辺の値を左辺へ代入することを示しています。ここでは説明の都合上，変数への代入は“：＝”で統一することにします。

代入文の構文は

となります。

代入文のコンパイル法を流れ図にすると**図3-12**の①のようになります。そして，この流れ図をプログラムにした場合，つくられるオブジェクト・プログラムは**図3-12**の②のようになります。

〔2〕単項演算

実際の式にはこれまで説明した二項演算の他に単項演算があります。単項演算には変数，定数や“(”の前につく“符号”，論理演算の“否定（NOT)”があり，これらを示す演算子を単項演算子といいます。

図3-12 代入文のコンパイル

①代入文のコンパイル法を示した流れ図

代入文

トークンは変数名？ No → エラー

Yes

※1

変数名のアドレスを求める … 〔求めたアドレスは一時的に記憶しておく

次のトークンを読む

式

式の演算結果をストアするオブジェクト・コードを生成 … 〔ストアするアドレスは※1で求めたアドレス

リターン

② ①の流れ図でつくられるオブジェクト・プログラム

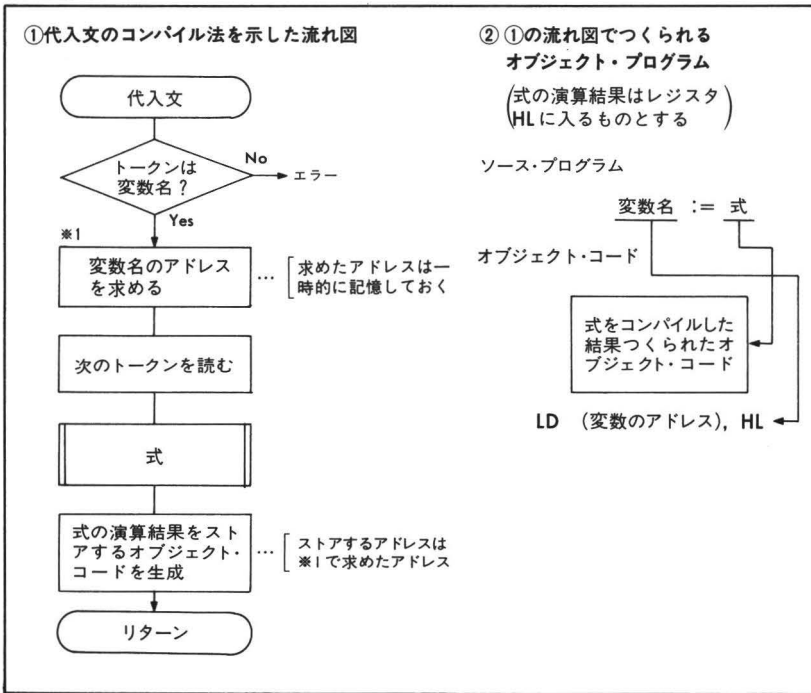
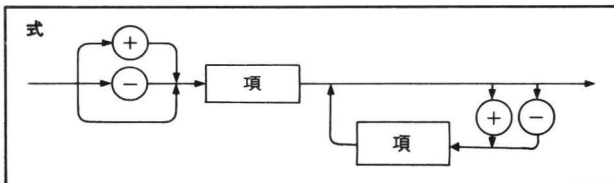
（式の演算結果はレジスタHLに入るものとする）

ソース・プログラム

変数名 := 式

オブジェクト・コード

式をコンパイルした結果つくられたオブジェクト・コード

LD (変数のアドレス), HL

ここまで例として使ってきた式の構文を単項演算子の符号“＋”“－”が使えるよう変更すると〈式〉の構文は次のようになります。

この構文図から，単項演算子の“＋”“－”は式の先頭，代入文でいえば“：＝”の直後，あるいは“(”の直後だけに現われることがわかります。

式を逆ポーランド記法に変換するときこの点に注意すれば，加減演算子“＋”“－”が単項演算子として使われているのか二項演算子として使われているのかが区別で

きます。そして，区別された加減演算子は区別できる形で出力します。ただし単項演算子の“＋”はあってもなくてもよいので出力しないようにします。ここでは単項演算子の“－”を“[－]”と表すことにすると，たとえば

```
−A*(−B+2)
```

という式は

```
A [−] B [−] 2+*
```

という逆ポーランド記法の式に変換されます。そして，この逆ポーランド記法の式を機械語に変換するときには，単項演算子［－］は符号を反転させるようなオブジェクト・コードを生成するようにします。

再帰的な方法で単項演算子を使った式をコンパイルするときは，たとえば式の構文が次のようになっていたら，この構文のとおりにプログラムをつくります。

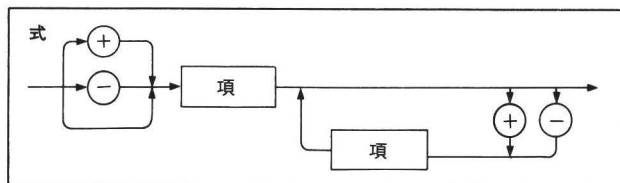



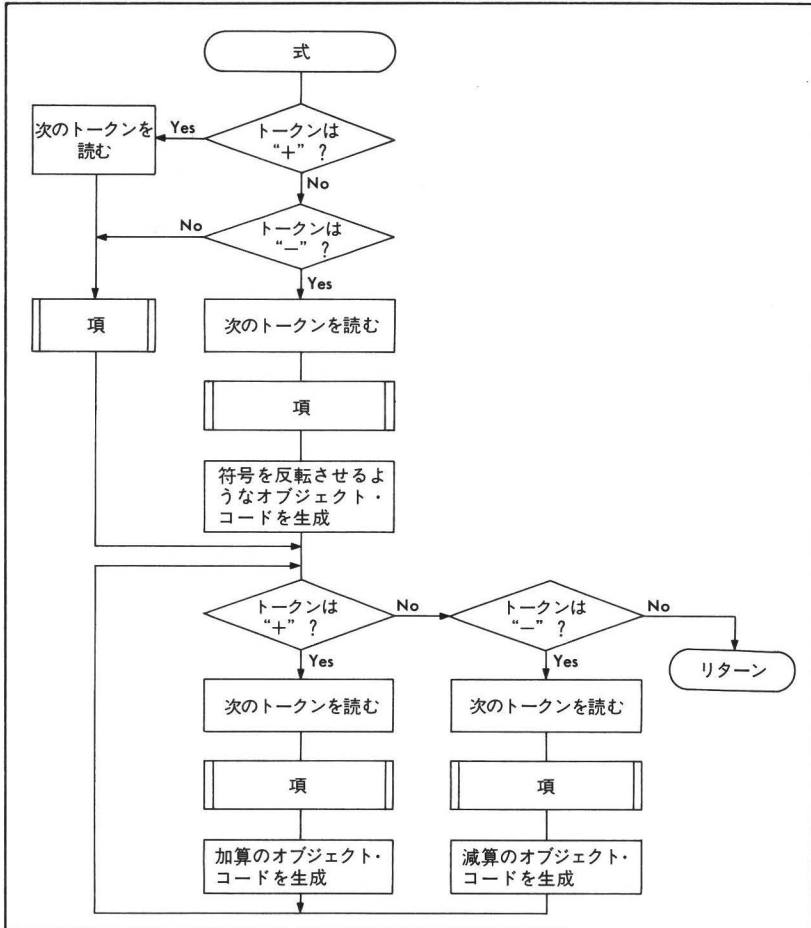
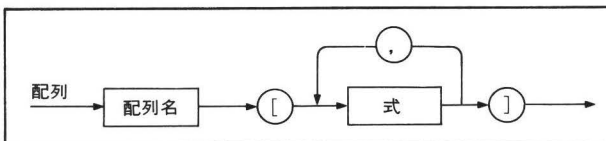
図3-13がこの式の構文からつくった流れ図です。この流れ図をプログラムにすれば単項演算子の符号が式の中で使えるようになります。

単項演算子には他にも論理演算の否定（NOT）がありますが，これも同様の方法でコンパイルできます。

〔3〕配列

式の中には変数や定数以外に配列や関数も書かれます。ここでは配列（添字つき変数ともいう）のコンパイル法について説明します。

図3-13 単項演算子"＋""－"が使える式を再帰的にコンパイルする

式
トークンは"＋"？
Yes
次のトークンを読む
No
トークンは"－"？
No
Yes
項
次のトークンを読む
項
符号を反転させるようなオブジェクト・コードを生成
トークンは"＋"？
No
トークンは"－"？
No
リターン
Yes
次のトークンを読む
項
加算のオブジェクト・コードを生成
Yes
次のトークンを読む
項
減算のオブジェクト・コードを生成

Pascal や ALGOL での配列の構文は次のようになっています。

BASICやFORTRANでは，［　］の代わりに（　）が使われています。ここでは説明の都合上［　］

に統一することにします。

また，一般的に配列はプログラムの先頭で次元数，添字の範囲などを宣言しなければ使えません。

配列を使った普通の式は

```
A [I+3] *B [J−1, K]
```

というように表わされますが，この式を逆ポーランド記法に変換すると

```
A [I 3+] B [J1−, K] *
```

となります。

式の中の配列を逆ポーランド記法に変換するとき，トークンの区切り記号“[　”“　]”“,”は次のように処理します。

●区切り記号“[　”

“[　”は逆ポーランド記法の式として出力するとともにスタックにもPUSHする。

●区切り記号“,”

スタック・トップが“[　”になるまで演算子をPOPし，出力する。その後“,”を出力する。

●区切り記号“　]”

スタック・トップが“[　”になるまで演算子をPOPし，出力する。その後“　]”を出力する。そして，スタック・トップの“[　”は必要なくなったのでPOPし捨てる。

基本的にはこの処理で配列を含んだ式を逆ポーランド記法に変換できます。実際にプログラムをつくる場合は，トークンの出現順序が正しいかどうかもチェックしなければなりません。

次に配列を含んだ逆ポーランド記法の式からオブジェクト・プログラムをつくります。構文的には変数名も配列名も関数名も同じで，変数名なのか配列名なのかは記号表を引くことでわかります。記号表から引いた名前が

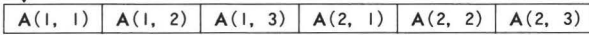
変数だったらこれまでと同じように変数の内容をロードしてくるようなコードを生成します。もし配列だったら記号表から配列の先頭アドレスや次元数，添字の範囲などの情報を取り出し，１次元配列なら**図3-14**の①のように，２次元以上の配列なら**図3-14**の②あるいは③のようにオブジェクト・コードを生成します。

図3-14 式の中の配列のオブジェクト・コード

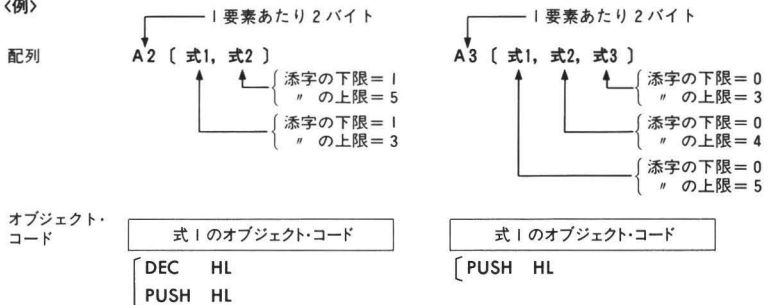
〈例〉
記号表より配列名であることを知る。1要素あたりのバイト数は2バイト，次元数は1，添字の下限は1とする。
1次元配列
AR1〔 式 〕
式のオブジェクトを生成
オブジェクト・コード
式のオブジェクト・プログラム
式の演算結果はレジスタHLに入いるものとする。
DEC HL
添字の下限が1なので演算結果より1減じている。
ADD HL, HL
1要素あたりのバイト数が2バイトなので，左へ1ビット・シフトするようなオブジェクト・コードを生成。
LD DE, 配列名のAR1の先頭アドレス
ADD HL, DE
参照したい配列要素のアドレスが求まる。
LD E, (HL)
INC HL
LD D, (HL)
EX DE, HL
レジスタHLに配列要素をロードする。
② 2次元以上の配列(第1の方法)
n次元配列
配列名〔 式1 , 式2 , ・・・ 式n 〕
1次元配列として見た場合
1次元配列
配列名〔(式1− ℓ1)× Π Ui + (式2− ℓ2)× Π Ui + ・・・ +(式n− ℓn)
オブジェクト・コード
式1のオブジェクト・コード
定数 ℓ1の減算
定数 Π Ui の乗算
式2のオブジェクト・コード
定数 ℓ2の減算
定数 Π Uiの乗算
加 算
式nのオブジェクト・コード
定数 ℓnの減算
加 算
配列1要素あたりのバイト数を乗算
配列の先頭アドレスを加算
加算結果が示すメモリの内容をロード

コンピュータの中では，n次元の配列も1次元の配列として記憶される。そのためn次元から1次元にする方法によってメモリ上での配列要素の並び方が異なる。この第1方法では，たとえばA(2, 3)の配列はメモリ上で

配列Aの先頭アドレス
↓

A(1, 1)	A(1, 2)	A(1, 3)	A(2, 1)	A(2, 2)	A(2, 3)

となる（ただし，添字の下限は1とする)。

注）ℓ_iは，i番目の添字の下限，
u_iは，i番目の添字の上限，
$\prod_{i=2}^{n} a_i$は，$a_2 \times a_3 \times \cdots \times a_n$ という式を示す。

③ 2次元以上の配列(第2の方法)
n次元配列 配列名〔 式1 , 式2 , ・・・ 式n 〕
1次元配列としてみた場合
1次元配列 配列名〔(式1−ℓ1)+(式2−ℓ2)×Π(i=1, 1) Ui+ ・・・ +(式n−ℓn)×Π(i=1, n−1) Ui
オブジェクト・コード
式1のオブジェクト・コード
定数ℓ1の減算
式2のオブジェクト・コード
定数ℓ2の減算
定数 Π(i=1, 1) Uiの乗算
加 算
式nのオブジェクト・コード
定数ℓnの減算
定数 Π(i=1, n−1) Uiの乗算
加 算
配列1要素あたりのバイト数を乗算
配列の先頭アドレスを加算
加算結果が示すメモリの内容をロード
この第2方法では，たとえばA(2, 3)の配列はメモリ上で
配列Aの先頭アドレス
A(1, 1)
A(2, 1)
A(1, 2)
A(2, 2)
A(1, 3)
A(2, 3)
となる（ただし，添字の下限は1とする）。
〈例〉
配列
1要素あたり2バイト
A2〔 式1, 式2 〕
添字の下限＝1
〃 の上限＝5
添字の下限＝1
〃 の上限＝3
1要素あたり2バイト
A3〔 式1, 式2, 式3 〕
添字の下限＝0
〃 の上限＝3
添字の下限＝0
〃 の上限＝4
添字の下限＝0
〃 の上限＝5
オブジェクト・コード
式1のオブジェクト・コード
DEC HL
PUSH HL
式1のオブジェクト・コード
PUSH HL

式 2のオブジェクト・コード

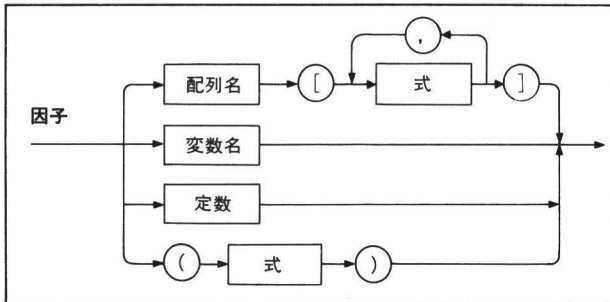
```
DEC     HL
LD      DE, 3
CALL    IMUL   ; HL←HL＊DE
POP     DE
ADD     HL, DE
ADD     HL, HL   ; HL←HL＊2
LD      DE, 配列A2の先頭アドレス
ADD     HL, DE
LD      E, (HL)
INC     HL
LD      D, (HL)
EX      DE, HL
```

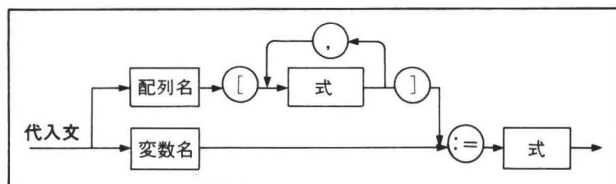
式 2のオブジェクト・コード

```
L D     DE, 5
CALL    IMUL   ; HL←HL＊DE
POP     DE
ADD     HL, DE
PUSH    HL
```

式 3のオブジェクト・コード

```
LD      DE, 5＊4
CALL    IMUL   ; HL←HL＊DE
POP     DE
ADD     HL, DE
ADD     HL, HL   ;HL←HL＊2
LD      DE, 配列A3の先頭アドレス
ADD     HL, DE
LD      E, (HL)
INC     HL
LD      D, (HL)
EX      DE, HL
```

再帰的な方法でコンパイルするときは，配列は変数と同じ〈因子〉で処理します。配列が使える因子の構文は

となり，変数名なのか配列名なのかは記号表を引いてわかります。〈因子〉の構文を流れ図にすると**図3-15**のようになります。オブジェクト・コードの生成は構文の解析結果から**図3-14**のようにします。

次は，代入文の左辺に書かれる配列についてです。左辺に配列が使える代入文の構文は，

となります。変数名か配列名かは記号表を引いて識別し，変数名なら〔1〕で説明したようなオブジェクト・コードをつくります。配列名なら式の中の配列と同様の方法で処理しますが，生成するオブジェクト・コードは**図3-16**のようにします。

ここで示した配列のオブジェクト・コードは，添字の範囲をチェックしていません。たとえば配列X（E）の添字の下限が1，上限が5なら変数Eの値が1～5以外なら本来はエラーとするようなオブジェクト・コードを生成したほうがよいのです。しかしそうするとオブジェクト・プログラムが大きくなって実行速度が落ちるため，パソコン用のコンパイラの多くは添字の範囲のチェックをしていません。

〔4〕関数の呼び出し

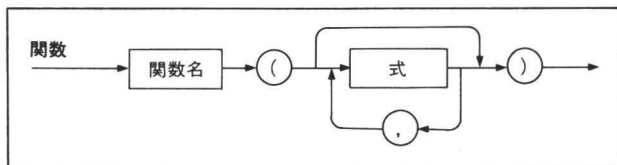
まず，式の中で使われる関数の構文を示します。

この構文は，［　］が（　）になり，カッコ内の式を書かなくてもよいことを除いては配列と同じであることがわかります。ですから，関数の構文解析は配列と同じようにつくることができます。ただし，この解析結果からつくられるオブジェクト・コードについては**3-7**で説明しますので，そちらをごらんください。

図3-15 配列が使用できる因子の流れ図(再帰的な手法)

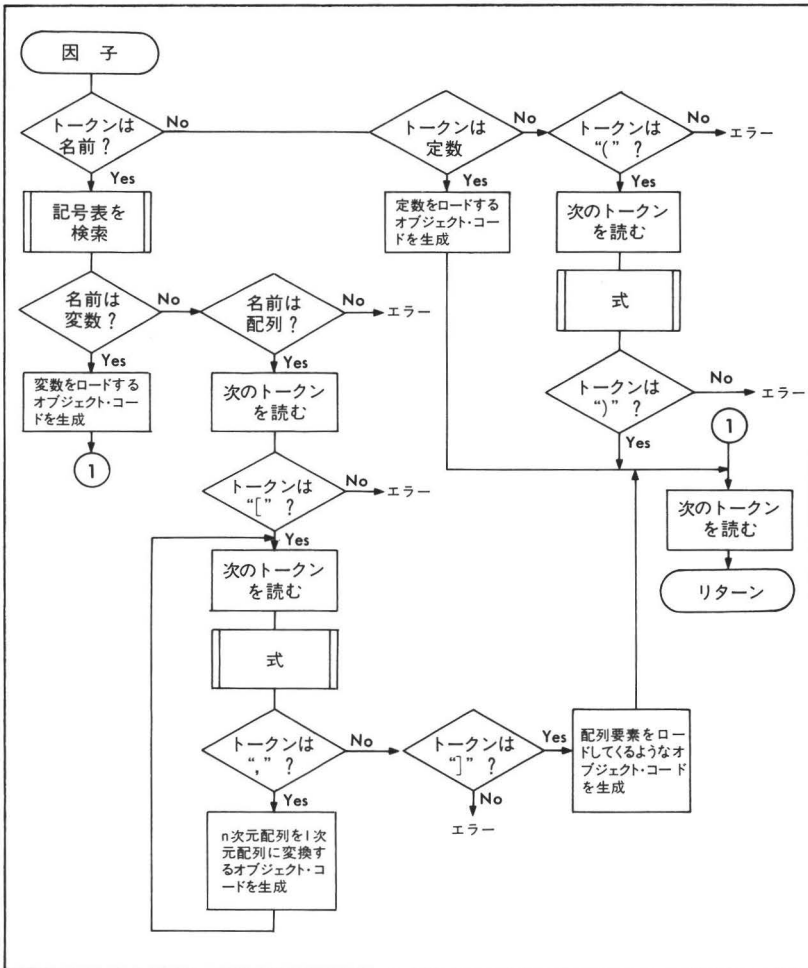
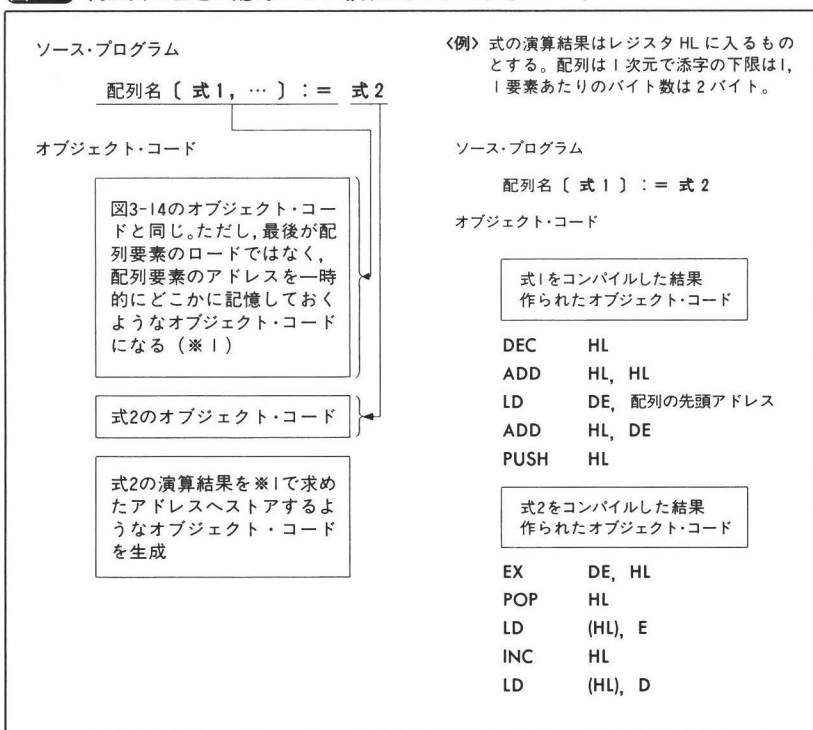



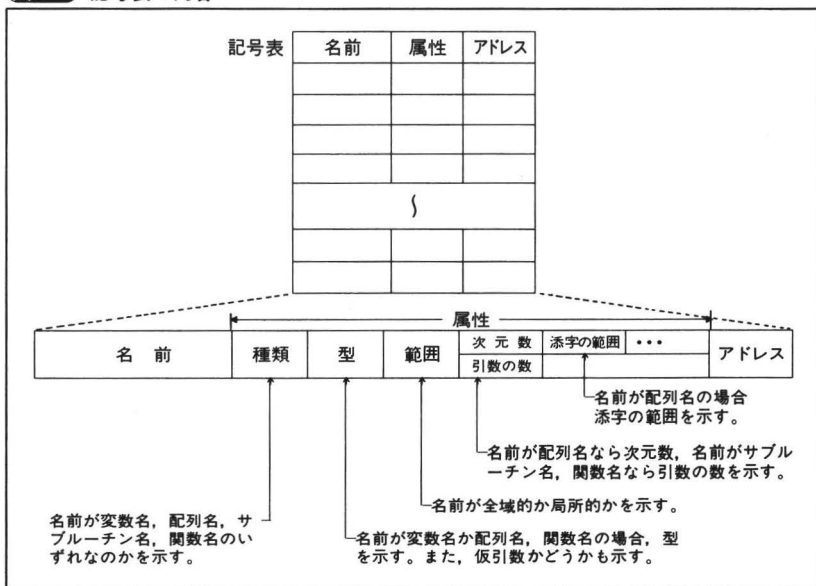
図3-16 代入文の左辺に配列がある場合のオブジェクト・コード

これで一通り式のコンパイルについて終わりますが，この項で示したオブジェクト・コードは構文解析の結果の通りに変換したものなので，効率の点で見劣りするかもしれません。実際にはオブジェクト・コードの生成順序をスタックを使って変えるなどを行い，できる限り効率化を図ってください。

3-4 記号表と宣言文

この項では，記号表の構成と管理，および変数名や配列名，型などを宣言する宣言文について説明します。

3-4-1 記号表の構成

コンパイラの中には名前（変数名，配列名，サブルーチン名，関数名，ラベルなど）とその属性（attributes）やアドレス（または値）を記憶しておく記号表という表があります。**図3-17**はその一例です。この中で属性はコンパイラによって項目数や内容が異なります。たとえば，変数，配列，関数の型が一種類しかないときは型の項目は必要ないでしょうし，コンパイラが再配置可能なオブジェクト・プログラムを出力するなら，名前が外部記号

図3-17 記号表の内容

であるのか内部記号であるのかを示す項目が必要になります。記号表はこれらを一要素とする１次元の配列としてメモリ上に記憶されます。

3-4-2 記号表の検索と登録

記号表の検索方法で代表的なものには，
①**シリアル・サーチ**（serial search :**逐次検索法**）
②**バイナリ・サーチ**（binary search : **2 分検索法**）
③**ハッシュ法**（hash）
などがあります。

シリアル・サーチは記号表を順に比較して検索する方法で，表に登録されている名前の数がｎ個なら，平均すると（ｎ＋１）/２回の比較で目的の名前が探し出されます。この方法はアルゴリズム（**図3-18**）が簡単である反面，ｎが大きくなるにつれて効率が悪くなります。登録も簡単で，新しい名前は記号表の最後に追加するだけです。

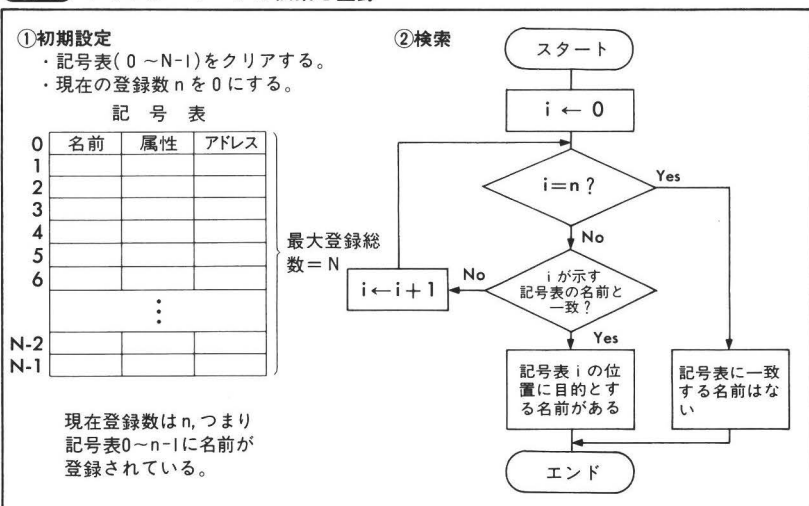
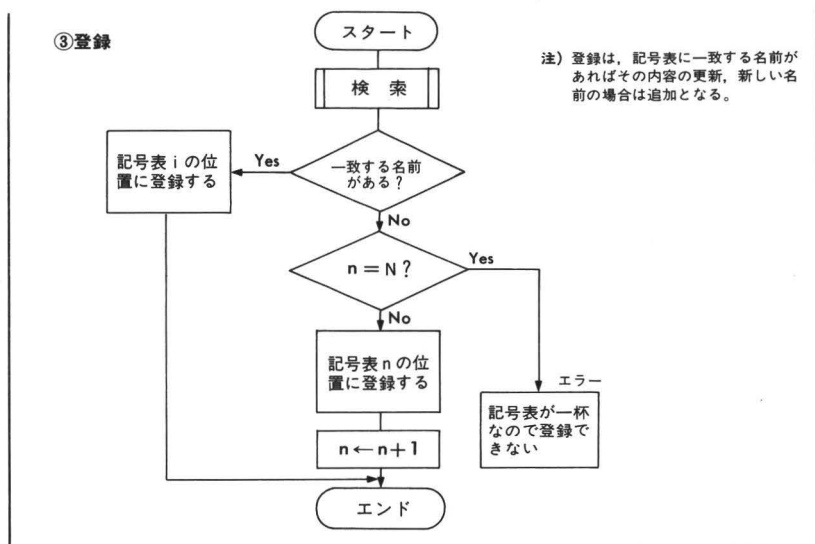
図3-18 シリアル・サーチの検索と登録

バイナリ・サーチは効率のよい検索方法ですが，表の名前が常に分類(sort：ソート)されていなければならないという欠点があります。この方法では，平均すると $\log_2 n$ 回の比較で検索できます。アルゴリズムは**図3-19**のようになっています。これは，まず初めに記号表の中央の名前（n/2番目にある名前）と比較します（一致すれば検索終了)。一致しなかったときには，表がソートされているため目的とする名前が中央より前のブロックにあるのか後のブロックにあるのかがわかります。そして目的とする名前が含まれていると思われるブロックに再び同じことをする，ということを繰り返して目的とする名前を探し出します(**図3-20**)。この方法では対象の表が常にソートされていなければならないことから，新しい名前を登録するには**図3-21**の①のようにする必要があります。この場合，表中のデータを実際にいちいち移動したのでは時間がかかりすぎるので，**図3-21**の②のように各要素の位置を示すポインタの表を用意します。新しい名前を登録するときは，記号表には単純に最後に加え，ポ

図3-19 バイナリ・サーチの検索と登録

①**初期設定**（図3-18の①と同じ）

②**検索**（記号表の名前は昇順に記憶されている）

スタート

j ← 0
k ← n-1

i←INT((j+k)/2)

i が示す記号表の名前と比較

記号表＝検索する名前 → 記号表 i の位置に目的とする名前がある

記号表＞検索する名前 → k←i－1

記号表＜検索する名前 → j←i＋1

j ＞ k　No → i←INT((j+k)/2) へ戻る

Yes → 記号表に一致する名前はない

エンド

③**登録**（昇順に登録する）

スタート

検索

一致する名前がある？　Yes → 記号表 i の位置に登録する

No

n＝N？　Yes → エラー：記号表が一杯なので登録できない

No

記号表j～n-1の内容をj+1～nへ転送

i ← j
n ← n＋1

記号表 i の位置に登録する

エンド

図3-20 バイナリ・サーチの検索例

図3-21 バイナリ・サーチのための記号表への新しい名前の登録

インタの表をソートします。こうすると常にポインタの移動だけで新規登録ができるので時間が節約できます。

ハッシュ法は最も効率のよい方法で，一回あるいは数回の操作で目的とする名前を探し出すことができます。ハッシュ法は“ハッシュ関数”という関数を用意して，鍵（記号表の場合は名前）をその関数から求めた値で登録位置とし，目的とするデータ（この場合，属性とアドレス）を探す方法です。ハッシュ関数にはいろいろなつくりかたがあります。ここでは例として名前を鍵，記号表の最大登録総数をＮとして

i ＝（名前の各文字を数として合計した値）mod N

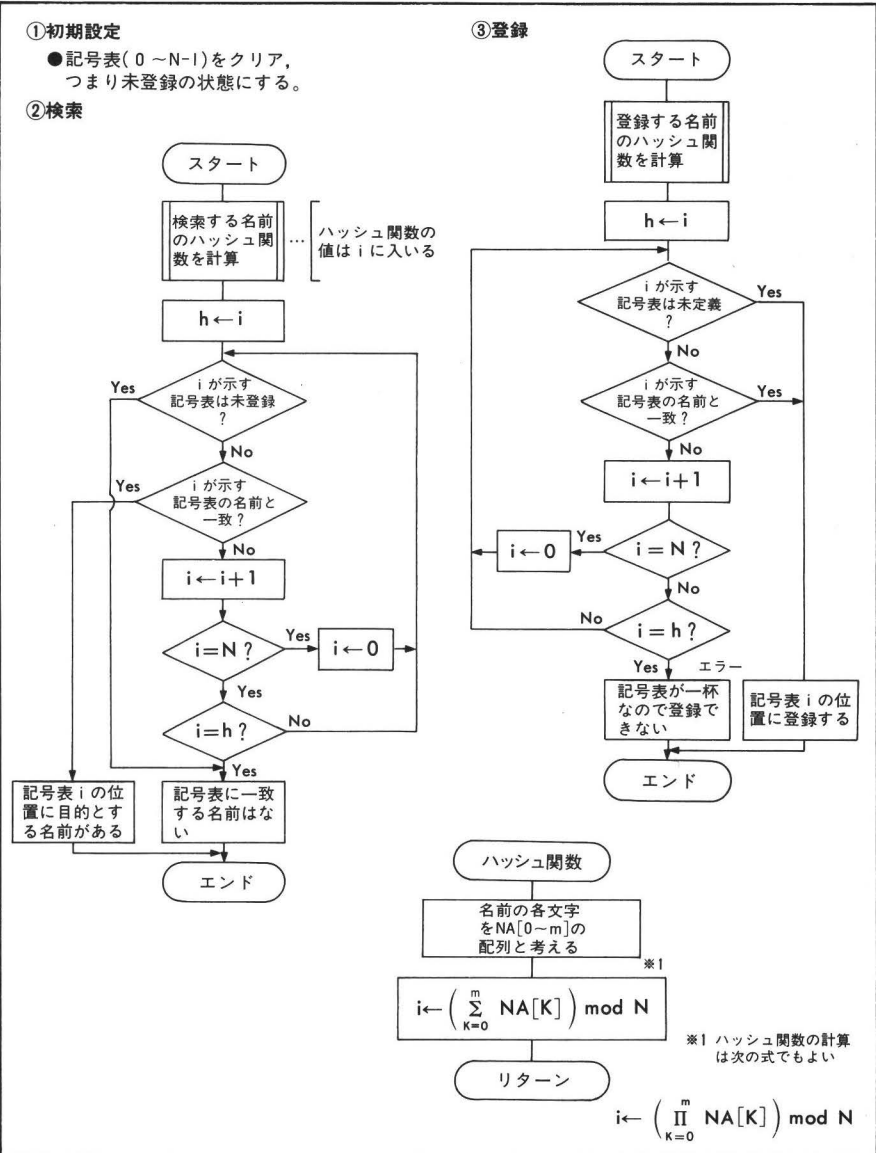
というハッシュ関数を考えます。“名前の各文字を数として合計した値”とは各文字のＡＳＣＩＩコードを足したものという意味です。この場合，ｉがハッシュ関数の値（０～Ｎ－１の値をとる）で記号表内の位置を示します。そして，登録，参照はこのｉの位置に行われます。たとえば，“ＡＢＣＤ”という名前があったとします。最大登録総数が50の記号表の場合，この名前の位置は

（“A”＋“B”＋“C”＋“D”） mod 50
＝（41H＋42H＋43H＋44H） mod 50
＝（65＋66＋67＋68） mod 50
＝ 16

と求められます。ただし，ここで注意しなければならないのは，このハッシュ関数では“ＤＣＢＡ”という名前でもｉが同じ値になってしまうということです。そこでこのような場合の対策として，登録時にはｉの位置から順に空いている位置を探しそこへ登録するようにします。検索時にはｉの位置からシリアル・サーチで目的の名前を探し出すようにします。このような方法を特にオープン・ハッシュ法といい，アルゴリズムを**図3-22**に示します。

小型のコンパイラで記号表の最大登録数が 500程度な

ら，検索速度やプログラム・サイズから考えて，アルゴリズムの簡単なシリアル・サーチが妥当だと思われます。

図3-22 オープン・ハッシュ法による検索と登録

3-4-3　パスと表管理

1パス方式と2パス以上の方式では記号表の管理のしかたが異なります。

〔1〕1パス方式

1パス方式では記号表への登録と検索を同時に行います。そのためソース・プログラム上で宣言される前に名前が使われることがあるという問題があります。このような場合，記号表から名前を検索しても未登録となってコード生成ができません。これは次のように解消します。

①記号表に名前が登録されていなかったとき

・構文から名前の種類を判断します。たとえば，配列なら名前の直後には“[　”があり，関数なら“(”，それ以外なら変数名とする，などと判断します。

・記号表の属性に未定義を表現するフラグがあるものとして，名前，属性（未定義フラグはON，他の属性は構文から判断した内容を設定），アドレス（ゼロにしておく）を登録します。そして，この内容を検索された名前の属性，アドレスとしてコード生成します。

②記号表には名前が登録されているが，属性の未定義フラグがON

・検索された内容でコード生成します。このとき，名前のアドレスを記憶した（または記憶する）アドレス（ロケーション・カウンタの値）を一時的にどこかに記憶しておきます（ここではTに記憶するとします）。

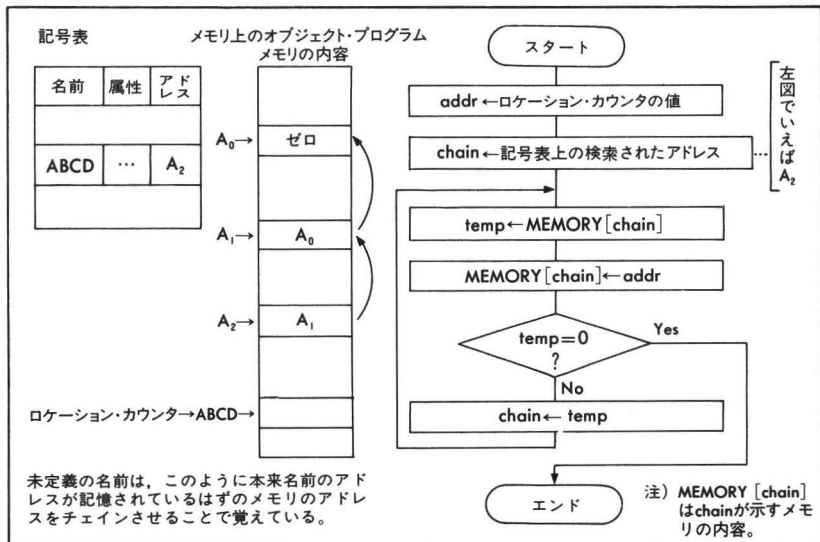
・記号表へ名前，属性，アドレス（Tの値）を再登録します。

③記号表に名前が登録されていて，属性の未定義フラグもOFF

・検索された内容でそのままコード生成します。

これらに伴い，宣言文などが新たに名前を登録するときには次のような処理をします。

①すでに記号表に名前が登録されているとき
・属性の未定義フラグがＯＦＦ，または属性の内容が宣言された内容と異なる場合はエラーとします。
・直接メモリ上に機械語をつくるコンパイラなら**図3-23**のように処理して，未定義のアドレスを現在のアドレス（ロケーション・カウンタの値）にします。オブジェクト・プログラムをファイルに出力するコンパイラなら，未登録を示す情報と名前のアドレスを出力します。ローダあるいはリンカでは，その情報を入力して直接メモリ上に機械語をつくるコンパイラと同じ方法で未定義のアドレスを正式な名前のアドレスに変えてやります。
・属性の未定義フラグをＯＦＦにし，アドレスを現在のアドレス（ロケーション・カウンタの値）にして再登録します。
②未登録な名前の場合
・宣言されている通りに属性を設定し（未定義フラグはＯＦＦ），現在のアドレス(ロケーション・カウンタの値)をアドレスとして記号表へ登録します。

図3-23 1パス方式の未定義アドレスの処理

〔2〕 2パス以上の方式

2パス以上の方式ではパス1で記号表を作成するので，パス2以降は記号表を参照するだけです。そのため，パス1とパス2以後では記号表に対する処理が次のように異なります。

①パス1

・記号表へは追加するだけで再登録はできません。もしすでに登録されている名前で登録しようとした場合はエラーとします。

・記号表を検索した結果，未登録なら構文からその名前の種類を判断し，記号表に一致する名前があったときと同じ処理をします。ただし，1パス方式のときのような登録は行いません。

・宣言文などで宣言されることによって初めて記号表への名前の登録を行います。

②パス2以後

・記号表は検索されるだけで登録はされません。

・記号表を検索した結果，未登録ならエラーとします。

・宣言文などの宣言は無視します。

このように1パス方式と2パス以上の方式では記号表の管理のしかたが異なり，それに伴い構文解析やコード生成といった部分も多少影響を受けています。ここで示した処理は記号表に対する基本的な処理なので，実際には言語やコンパイラの実行環境によって処理が多少複雑になることがあります。

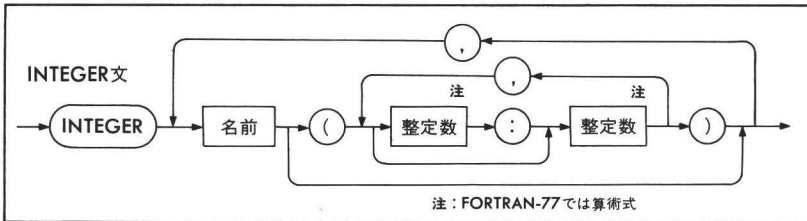
3-4-4 宣言文の処理とラベル

プログラム言語では宣言文やラベルというものが使われます。宣言文やラベルと記号表とは大変深い関係にあるため，この項で宣言文の処理とラベルの説明をすることにします。

宣言文には大きく分けてデータに関する宣言と実行制御に関する宣言の二種類があります。

〔1〕データに関する宣言

これはプログラム中で使う変数や配列の名前の宣言です。コンパイラは宣言された名前に対して属性を決め，アドレスを割り当て記号表へ登録しプログラム中でその名前が使えるようにします。

たとえばBASICではDIM，FORTRANではDIMENSION，INTEGER，REALなどがあります。この中でFORTRANのINTEGER，つまり整数型であることを宣言するための文の構文は

となります。この構文図からつくった流れ図が**図3－24**です。

他の宣言文，あるいはFORTRAN，ALGOL系の他の言語の宣言文も基本的には同じような構造をしています。

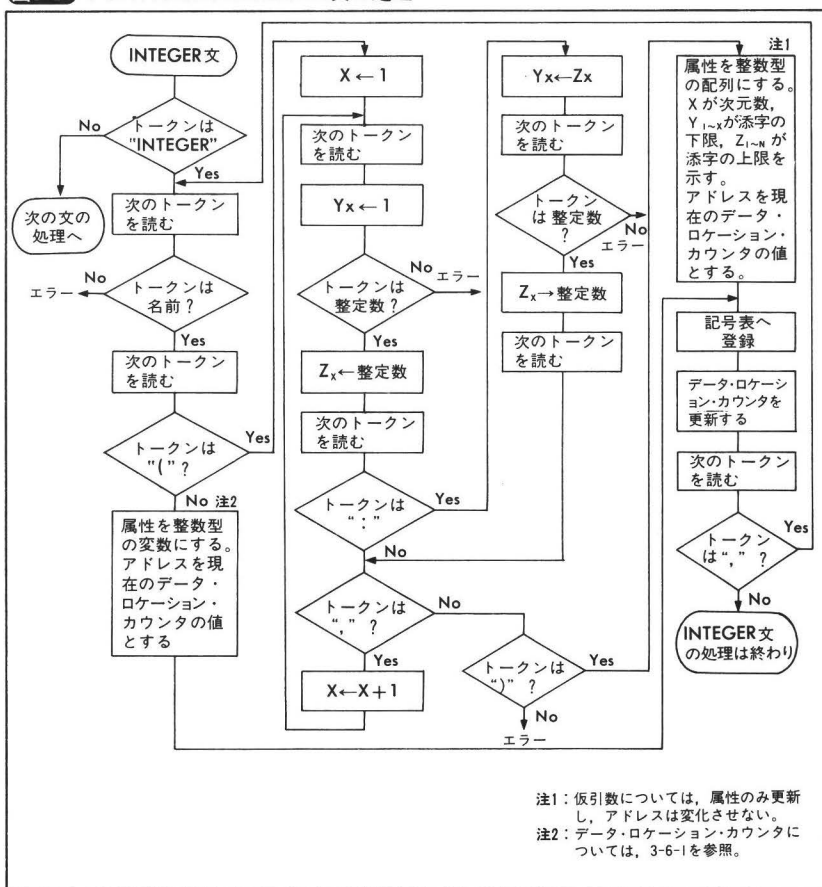
〔2〕実行制御に関する宣言

これはプログラム中で使うサブルーチンや関数の名前の宣言です。

同一の言語なら，サブルーチンと関数はトークンが一部異なるだけで構文的にはほぼ同一であることが多いようです。たとえば，Pascalではサブルーチン(Pascalでは手続きと呼ぶ）を

```
procedure　名前（仮引数，…）；
```

関数を

図3-24 FORTRANのINTEGER文の処理

function　名前（仮引数，…）：型；

というように表しますし，FORTRANではサブルーチンを

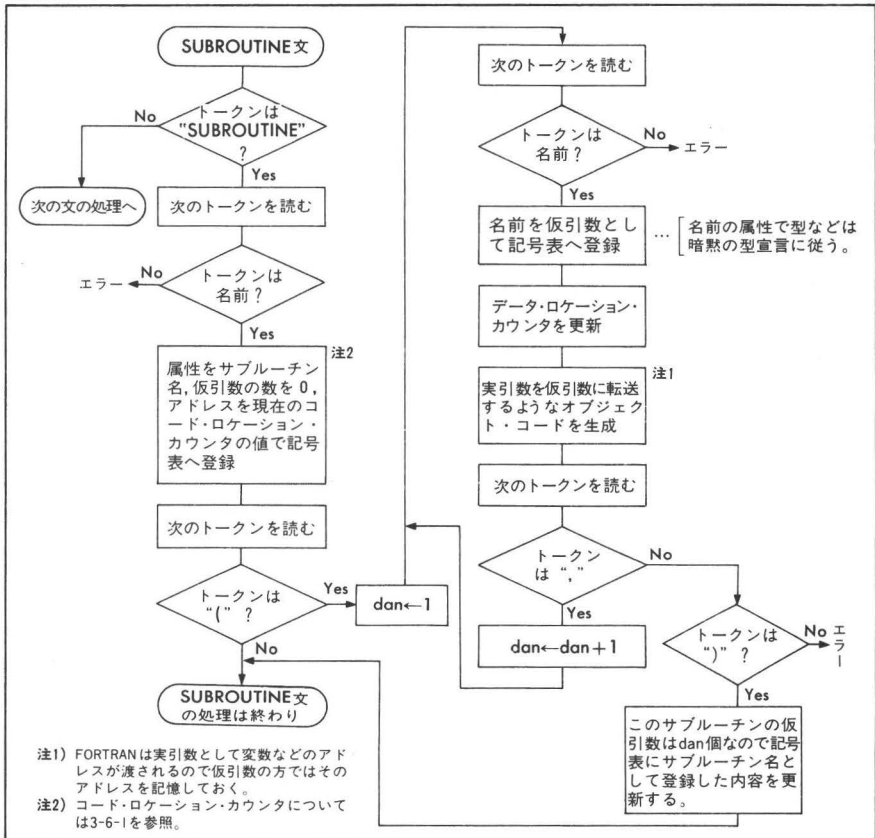
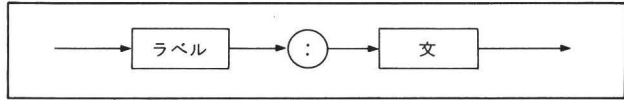
SUBROUTINE　名前（仮引数，…）

関数を

型　FUNCTION　名前（仮引数，…）

というように表します。ですから，コンパイラ内での実行制御に関する宣言の処理はどれもほぼ同じ構造でつくることができます。

これらの宣言は変数や配列の宣言とはちがい，名前を記号表に登録する以外に実引数を仮引数へ転送するようなオブジェクト・コードを生成します。同時に仮引数の名前と属性を決め，アドレスを割り当て記号表へ登録します。実引数を仮引数へ転送するオブジェクト・コードは引数を渡す方法によって異なります。詳しくは 3-7 で説明します。

図3-25 FORTRANのSUBROUTINE文の処理

次に，ラベルは文の前に書きその文に名前をつけるもので，制御文の飛び先として使われます。PascalやＡＬＧＯＬ，ＰＬ/Ｉでは

のように書きます。FORTRANやCOBOLなどではある特定の桁位置に書かれた数字または名前をラベルとしています。コンパイラはラベルに出会うと，その名前にアドレスを割り当て記号表に登録します。

最後に，宣言文によらず，式の中で新しい変数名や配列名を使えば自動的にコンパイラのほうが宣言してくれるような言語（BASICやFORTRANなど）についてです。このような言語のコンパイラは，式から名前を検索する際，次のような処理をします。

①検索した結果，一致する名前がなければ構文から属性を判断し，アドレスを割り当てて記号表へ登録する。

②登録した内容を検索した結果とする。

こうして変数や配列の自動宣言ができます。

3-4-5 全域的な名前と局所的な名前の処理

名前には有効範囲があります。たとえばFORTRANでは，メイン・ルーチン内の変数Xとサブルーチン内の変数Xとはまったく別のものとして扱われます。また，Pascalなどブロック構造を持つ言語では，サブルーチンや関数が書かれた（宣言された）場所によって，使える名前の有効範囲が異なります。ここでは，有効範囲を持った名前を記号表内でどう管理するかについて述べます。

BASICのようにどこでも同じ名前を共通に使う言語では，名前の使用範囲を管理する必要がありません。

FORTRANの場合，サブルーチン名や関数名（文関数ではなく関数副プログラムのこと），ブロック名などは

同一プログラム内では共通に使えます。つまり，サブルーチン名や関数名は全域的な名前であるわけです。ところが変数名や配列名などは一つのルーチン内（メインやサブルーチン，関数）だけで使い，他のルーチンからは使えません。逆にいえば，他のルーチンで同じ名前の変数や配列が使えるということです。つまり，変数名や配列名などは局所的な名前であるわけです。

このような言語では，全域的な名前と局所的な名前のために別々の記号表を用いるのも一つの手でしょう。**図3-26**が二つの記号表を用いた場合で，①が１パス方式の場合，②が２パス以上の場合です。１パス方式の場合，一つのルーチン（メイン，サブルーチン，関数など）のコンパイルが終わるごとにすべての名前を削除するようにすれば，局所的な名前の記号表が小型でも多くの名前を扱うことができます。２パス以上の方式では，パス２以後に渡すためにパス１でプログラム内の名前をすべて記号表に登録せねばなりません。そこで，図のように全域的な名前の記号表に局所的な名前の記号表に対するポインタをつけるようにし，局所的な名前どうしが干渉しないようにしています。

図3-26 全域，局所の二つの記号表を用いた例(FORTRAN)

①１パス方式

全域的な名前を記憶する記号表

名前　属性　アドレス

登録

この表にはサブルーチン名，関数名，ブロック名などが記憶される。

一つのルーチンのコンパイルが終わると局所的な名前を記憶している記号表の内容をすべてクリアする。この表には変数名，配列名が記憶される。

② 2パス以上の方式

次はPascalのようにブロック構造を持った言語の場合です（**図3-27**はブロック構造で名前が使える範囲を示した例）。コンパイラが1パス方式なら記号表をスタックのような構造にして処理します。**図3-28**の①がそのときの記号表の操作を示したものです。検索法としてシリアル・サーチを使うため検索効率があまりよくありませんが，方法が簡単なのでプログラミングが楽です。

2パス以上の方式では**図3—28**の②のように記号表をリスト構造にして処理します。パス1で一つのプログラム内で使われたすべての名前をリスト構造で登録し，パス2以後の検索では現在コンパイルしているブロックからポインタをたどって目的の名前を探し出すようにします。

2パス以上のときはパス1ですべての名前を記号表に登録すると書きましたが，実際にすべての名前を一度に管理するのはメモリ・サイズの増大や検索，登録ルーチ

ンの複雑化など問題が多いのです。特にブロック構造の場合は大変です。そこで，パス1ではパス2以後で最低必要な名前だけを登録するだけにして，他の名前はパス2以後でも登録，検索をするようにすれば，多少管理も簡単になります。ときにはこのような変則的な方法も必要です。

図3-27 ブロック構造と名前の有効範囲

ブロック内で宣言されたもの	それを使用することができるブロック
X	Y, A, B, Z, M, N
Y	Y, A, B
A	A
B	B
Z	Z, M, N
M	M, N
N	N

レベル0：X
レベル1：Y, Z
レベル2：A, B, M
レベル3：N

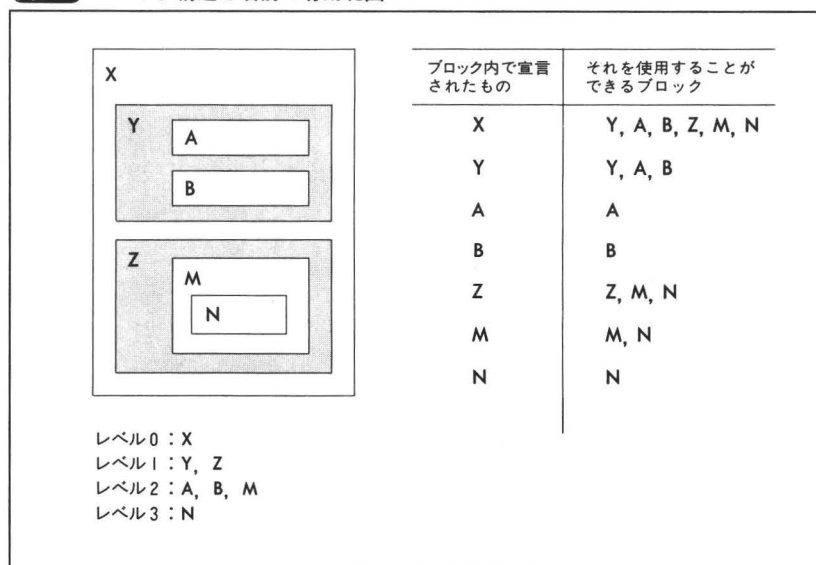
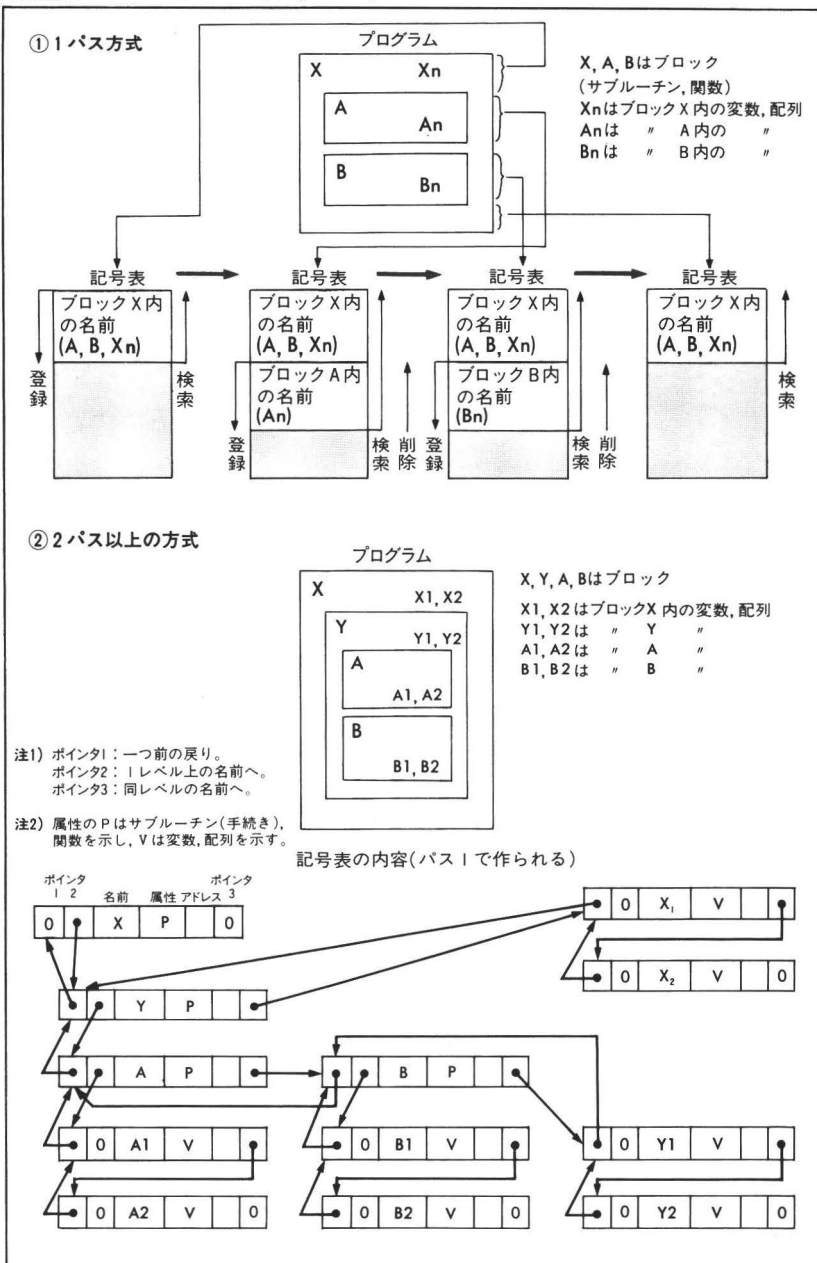



図3-28 ブロック構造と記号表(Pascal)

3-5 制御文のコンパイル

コンパイラの中で式と並んで大きなウェイトを占めているのが制御文の処理です。制御文にはIF文，GOTO文，FOR文，DO文，CALL文などの古典的な文や，WHILE文，REPEAT文，CASE文など新しい構造化プログラミングのための文などがあります。ここではこれらを一通り説明するために，多種の制御文を持ったPascal言語を例として，どうコンパイルし，オブジェクト・コードを生成するかを説明します。

3-5-1 条件文

条件文は流れ図記号の〝判断〟に相当する文で，PascalではIF文とCASE文の二つがあります。IF文はどのような言語にも必ず備わっている文です。CASE文は，古い言語ではないものが多いようです。

〔1〕IF文

IF文の構文は**図3－29**の①のようになっています。これを流れ図で表したのが②です。さらに，この流れ図をアセンブラ言語で表したのが**図3－30**です。コンパイラが①の構文から**図3－30**のオブジェクト・コードを生成するとき，一つ問題なのが文1や文2を飛び越す命令のジャンプ先アドレス（図ではラベルad1，ad2）の設定のしかたです。このアドレスは文1や文2のオブジェクト・コードの大きさがわからないうちは決められません。そこで，とりあえずアドレス部をゼロ（たとえば〝JP 0000H〞など）として仮のコードを生成しておき，文のオブジェクト・コードの生成が終わった時点で正式のアドレスを設定してやります。このことを加味したのが，**図3－31**の

オブジェクト・コードと図3－32のコンパイル手順の流れ図です。

このことは他の制御文にもある問題なので，同じように処理します。

図3-29 IF文(Pascal)

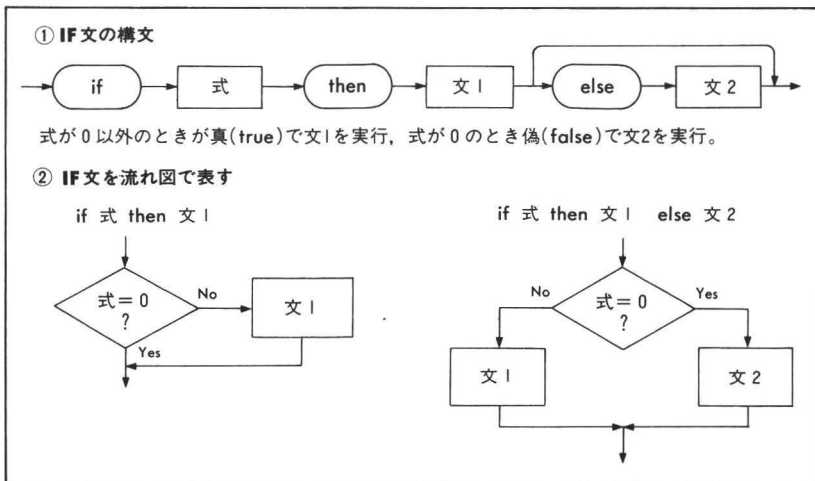



図3-30 IF文をアセンブラ言語で表す

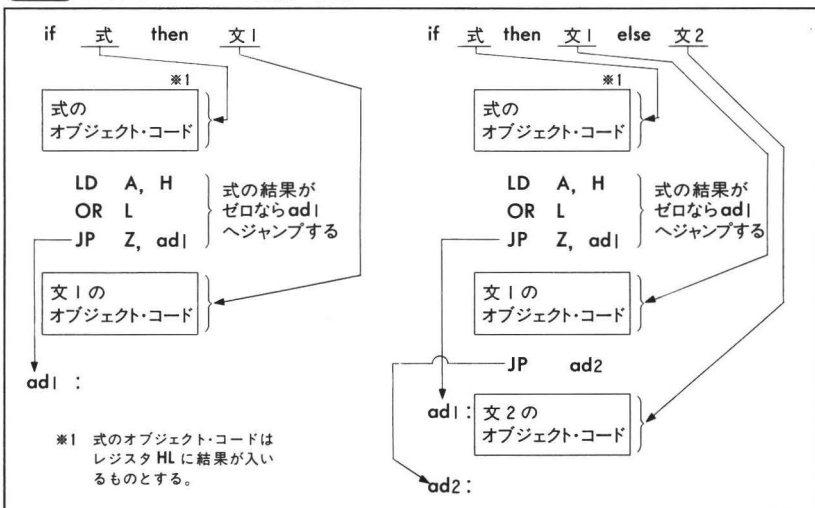



図3-31 IF文のオブジェクト・コード

ソース・プログラム　**if 式 then 文1**

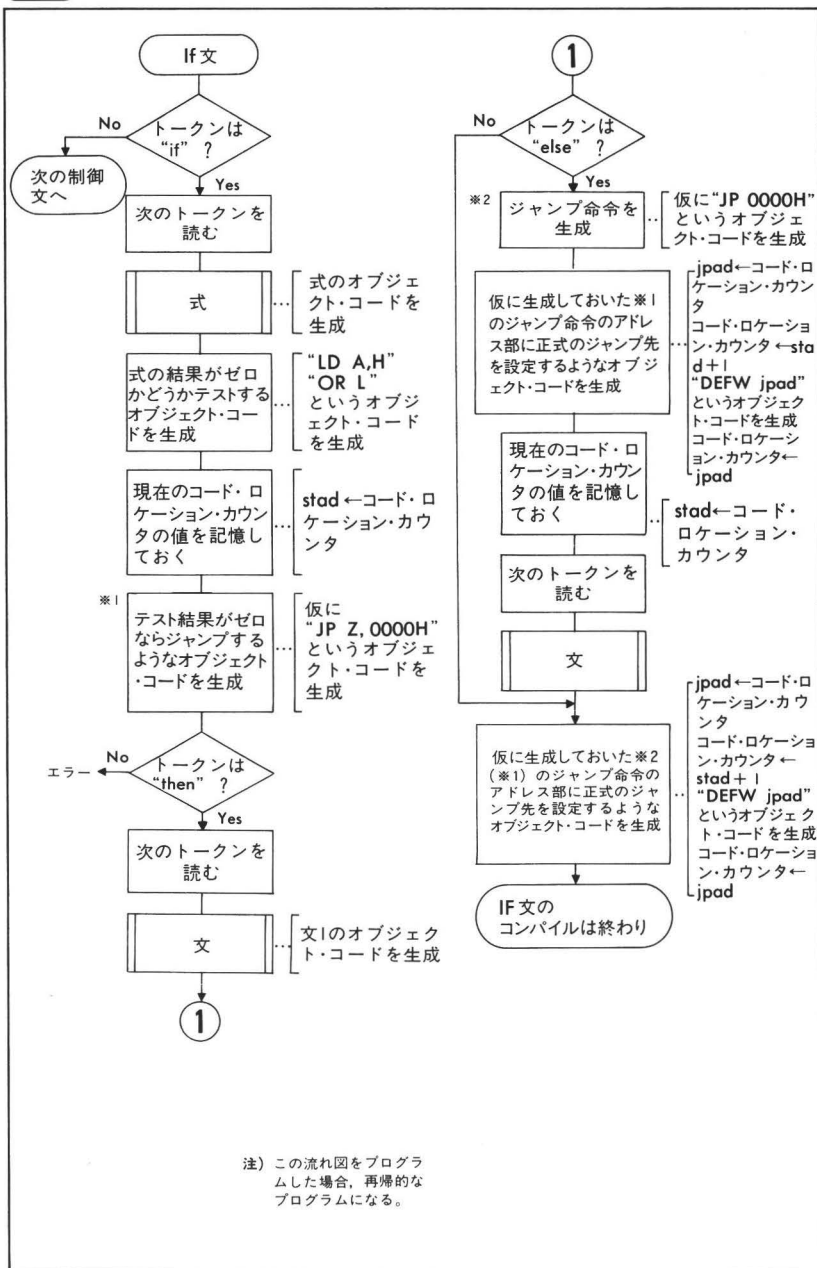
オブジェクト・コード

```
        [式のオブジェクト・コード]  } 結果はレジスタ HL に入いる
        LD      A, H
        OR      L
stad1:  JP      Z, 0000H            } 仮のコードを生成しておく
        [文1のオブジェクト・コード]
jpad1   EQU     $                   ┐ 仮に生成しておいたジャンプ
        ORG     stad1+1             │ 命令のアドレス部に正式のジ
        DEFW    jpad1               │ ャンプ先を設定
        ORG     jpad1               ┘
```

ソース・プログラム　**if 式 then 文1 else 文2**

オブジェクト・コード

```
        [式のオブジェクト・コード]
        LD      A, H
        OR      L
stad1:  JP      Z, 0000H
        [文1のオブジェクト・コード]
stad2:  JP      0000H
jpad1   EQU     $
        ORG     stad1+1
        DEFW    jpad1
        ORG     jpad1
        [文2のオブジェクト・コード]
jpad2   EQU     $
        ORG     stad2+1
        DEFW    jpad2
        ORG     jpad2
```

図3-32 IF文のコンパイル

〔2〕CASE文

CASE文の構文は図3－33の①のようになっています。これを流れ図で表したのが②，アセンブラ言語で表したのが図3－34です。図3－34は式の演算結果と定数との比較にランタイム・ルーチンを使っています。直接比較すると定数が多くなるにつれて比較部分の機械語が多くなります。たとえば，二つの定数と比較する場合，直接比較だと

```
        LD      HL, 定数1
        OR      A
        SBC     HL, DE
        JR      Z, EQCONS
        LD      HL, 定数2
        OR      A
        SBC     HL, DE
        JP      NZ, CASEX
EQCONS:
```

図3-33 CASE文(Pascal)

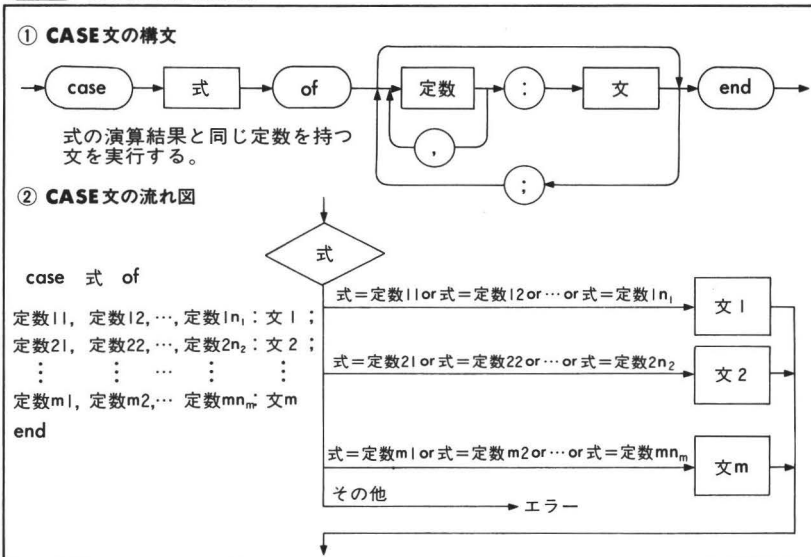
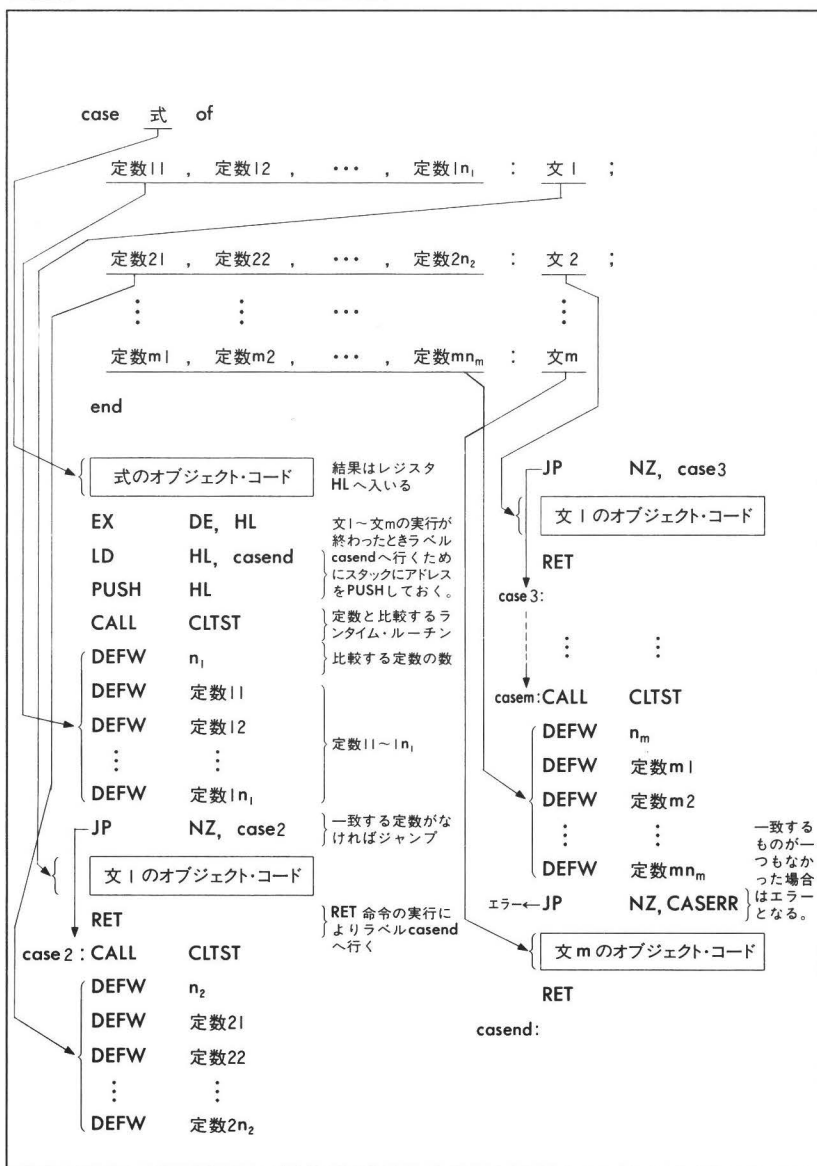



図3-34 CASE文をアセンブラ言語で表す

＊使用ランタイム・ルーチン
- ●CLTST…レジスタDEと指定された複数の定数と比較し，一致するものがあればZフラグを1に，なければZフラグを0にする。
- ●CASERR… エラーメッセージの表示とアボートを行う。

となり合計14バイトの大きさになります。ランタイム・ルーチンにCASE文用の定数比較ルーチンをつくっておくと

```
CALL   CLTST
DEFW   2
DEFW   定数1
DEFW   定数2
JP     NZ, CASEX
```

となり合計9バイトで，5バイト小さくなっています。このバイト数の差は比較回数につれて大きくなります。ところが，定数が一つの場合はどうでしょう。直接比較すると

```
LD     HL, 定数
OR     A
SBC    HL, DE
JP     NZ, CASEX
```

となり合計6バイトの大きさになります。ランタイム・ルーチンを使うと

```
CALL   CLTST
DEFW   1
DEFW   定数
JP     NZ, CASEX
```

と合計7バイト，今度は1バイト増えてしまいました。

CASE文の場合，ある文の条件になる定数は一文あたり一つの場合が多いようです。つまり，

```
case  式  of
      定数1  :  文1;
        :        :
      定数m  :  文m
end
```

と書かれることが多いわけです。ということは，直接比較したほうがランタイム・ルーチンで比較するより生成されるオブジェクト・コードが小さくなります。

以上のようなことから，CASE 文をコンパイルする場合，トークン〝：〞の前の定数が一つなら直接比較，定数が二つ以上ならランタイム・ルーチンを使うようなオブジェクト・コードを生成するようにします。こうしてCASE 文をコンパイルしたとき生成するオブジェクト・コードは図3－35のようにします。図3－36は，このようなコードを生成するCASE 文のコンパイル手順を流れ図にしたものです。

図3-35 CASE文のオブジェクト・コード

図3-36 CASE文のコンパイル

①
オブジェクト・コードとして "CALL CLTST" "DEFW n" を生成する。
nは定数の数
スタックより定数をPOP
オブジェクト・コードとして "DEFW 定数" を生成する。
実際には定数1～nは，DEFW 定数n ⋮ DEFW 定数1 と逆順に記憶される
n←n－1
n＝0
No
Yes
②
現在のコード・ロケーション・カウンタの値を記憶しておく
stad1←コード・ロケーション・カウンタ
※2
Z フラグが0ならジャンプするようなオブジェクト・コードを生成
仮に "JP NZ, 0000H" というオブジェクト・コードを生成
次のトークンを読む
③
③
文
"RET" のコードを生成
トークンは "end"?
Yes
⑤
No
トークンは ";"?
No
エラー
Yes
次のトークンを読む
トークンは定数?
No
Yes
仮に生成しておいた※2のジャンプ命令のアドレス部に正式のジャンプ先を設定するようなオブジェクト・コードを生成
jpad←コード・ロケーション・カウンタ
コード・ロケーション・カウンタ←stad 1＋1
"DEFW jpad"というオブジェクト・コードを生成
コード・ロケーション・カウンタ←jpad
④
⑤
仮に生成しておいた※1,.※2のジャンプ先を設定するようなオブジェクト・コードを生成
jpad←コード・ロケーション・カウンタ
コード・ロケーション・カウンタ←stad1＋1
"DEFW CASERR"というオブジェクト・コードを生成
コード・ロケーション・カウンタ←stad0＋1
"DEFW jpad"というオブジェクト・コードを生成
コード・ロケーション・カウンタ←jpad
⑥
次のトークンを読む
CASE文のコンパイルは終わり
注）この流れ図をプログラムした場合，再帰的なプログラムになる。

3-5-2　繰り返し文

繰り返し文は同じ文を繰り返し実行させる文で，Pascal には WHILE 文，REPEAT 文，FOR 文の 3 種類あります。

〔1〕WHILE文

WHILE 文の構文は**図3－37**の①のようになっています。この文を流れ図で表したのが②です。WHILE 文は，条件が成立している間ある文を繰り返し実行する文で，初めに条件（式の値）を調べるので，条件によっては一度も文が実行されない場合もあります。この WHILE 文をアセンブラ言語で表したのが**図3－38**です。そして，**図3－39**が WHILE 文をコンパイルしたときのオブジェクト・コードです。**図3－40**はその生成手順を流れ図で表したものです。

WHILE 文の処理で一つ注意する点は，繰り返すためのジャンプ命令の生成です。IF 文のときとはちがい，ジャンプ命令を生成する段階でジャンプ先アドレスがわかっているので，相対ジャンプ命令で届く範囲であるかどうかもわかります。そこで，オブジェクト・コードをなるべく小さくするために，できるだけ2バイトで済むJR命令を生成するようにします。このことは WHILE 文以外でも同様です。

図3-37　WHILE文（Pascal）

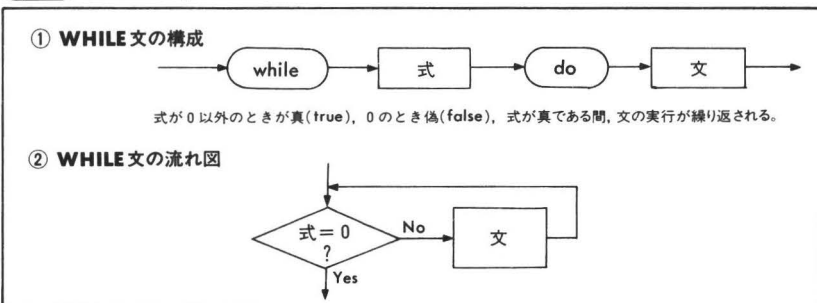



図3-38 WHILE文をアセンブラ言語で表わす

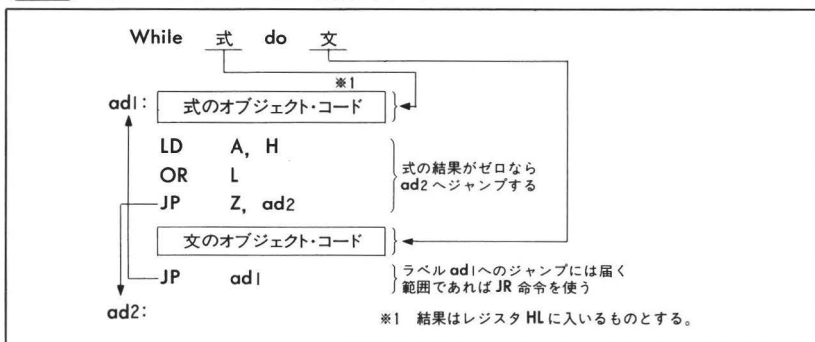
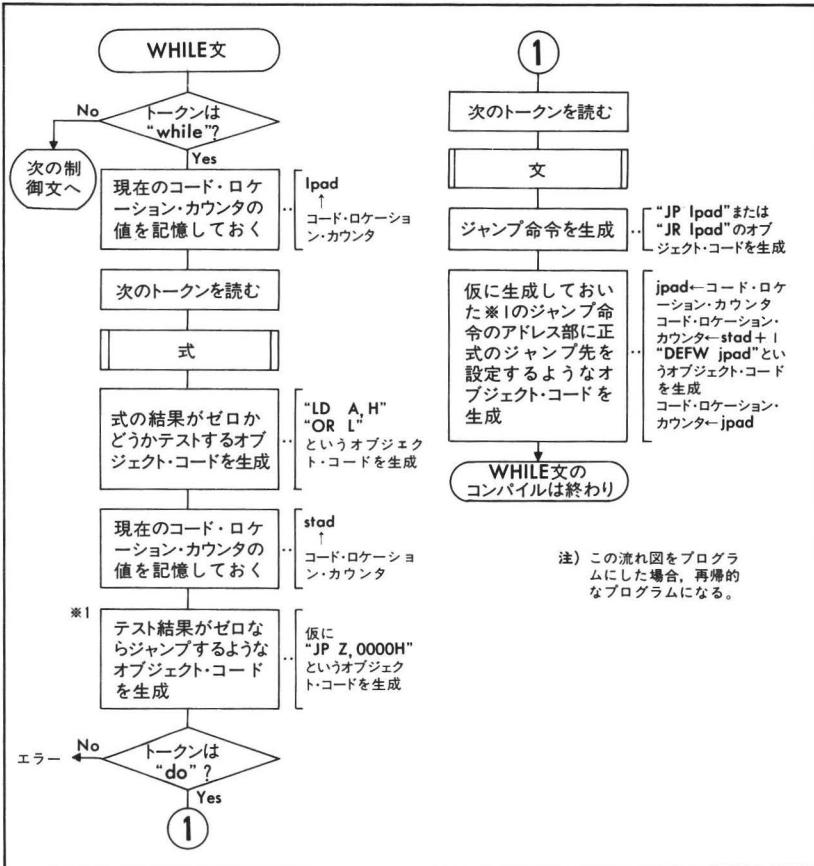
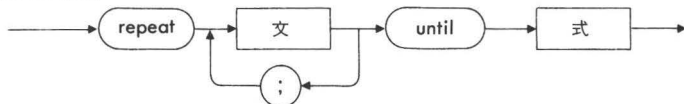
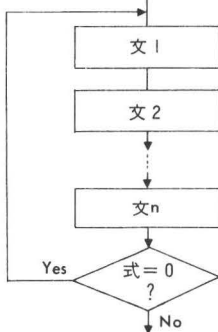



図3-39 WHILE文のオブジェクト・コード

図3-40 WHILE文のコンパイル

〔2〕REPEAT文

REPEAT 文の構文は**図3－41**の①のようになっています。この文を流れ図で表したのが②です。REPEAT 文は，ある条件が成立するまで，repeat と until の間にある文を繰り返し実行する文です。条件（式の値）があとで調べられるので，最低一度は repeat と until の間にある文が実行されます。これをアセンブラ言語で表したのが**図3－42**で，生成するオブジェクト・コードも同じです。**図3－43**はその生成手順を流れ図で表したものです。

図3-41 REPEAT文(Pascal)

REPEAT文をアセンブラ言語で表す(生成するオブジェクト・コードも同じ)

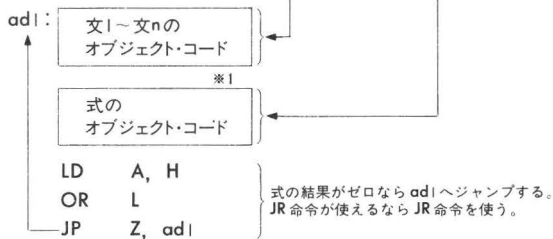
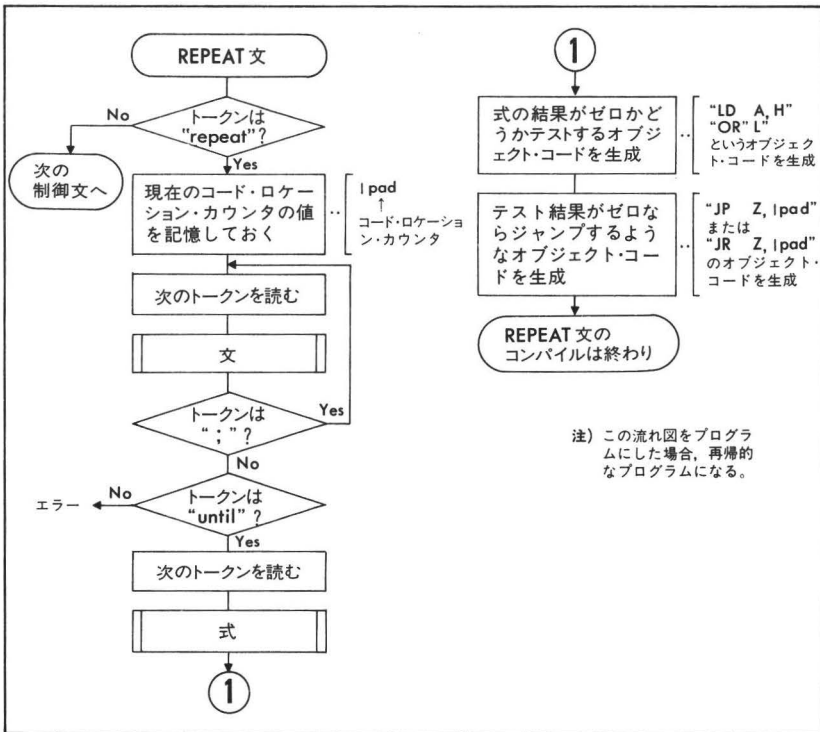
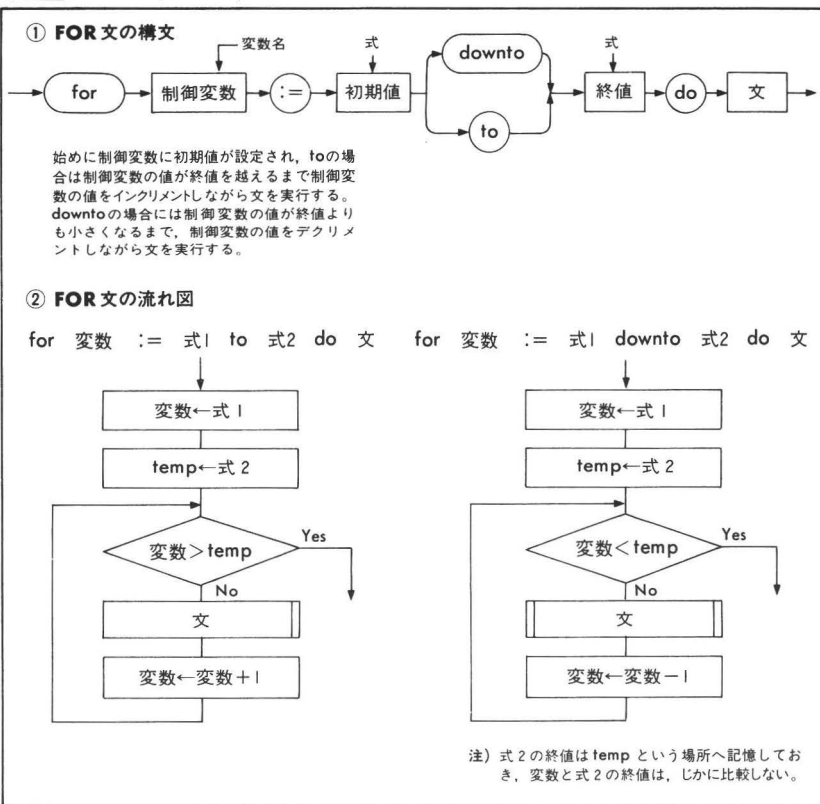



図3-43 REPEAT文のコンパイル

〔3〕FOR文

PascalのFOR文の構文は図3－44の①で，②がこの文を流れ図で表したものです。初期値と終値を示す式は繰り返しの前に計算され，初期値は制御変数に，終値はメモリ上に記憶されます。最初に制御変数とメモリ上の終値とを比較し，終了条件が成立していれば繰り返しを終わり，成立していなければ文を実行して制御変数の値を更新します。初期値と終値によっては文が一度も実行されません。終値の前に置かれたトークンが〝to〞だと繰り返し終了条件が

制御変数 ＞ 終値

となり，制御変数は文が実行し終わるごとにインクリメ

図3-44 FOR 文(Pascal)

ント（＋1）されます。〝downto〟だと繰り返し終了条件が

制御変数 < 終値

となり，制御変数はデクリメント（－1）されます。図**3－45**はFOR 文をアセンブラ言語で表したものです。図**3－46**が生成するオブジェクト・コードで，**図3－47**はその生成手順を流れ図で表したものです。

FOR 文をコンパイルするときの注意は，終値を変数と同じようにメモリ上に，しかも一つのFOR 文ごとに一つずつ記憶することです。たとえば一つのプログラムに

FOR 文が100あったとすると，終値を2バイトで表すものとして，2バイト×100=200バイトの終値記憶場所をメモリ上に確保します。わざわざメモリ上に記憶するのは，ループ中に終値の計算を入れないようにするためで

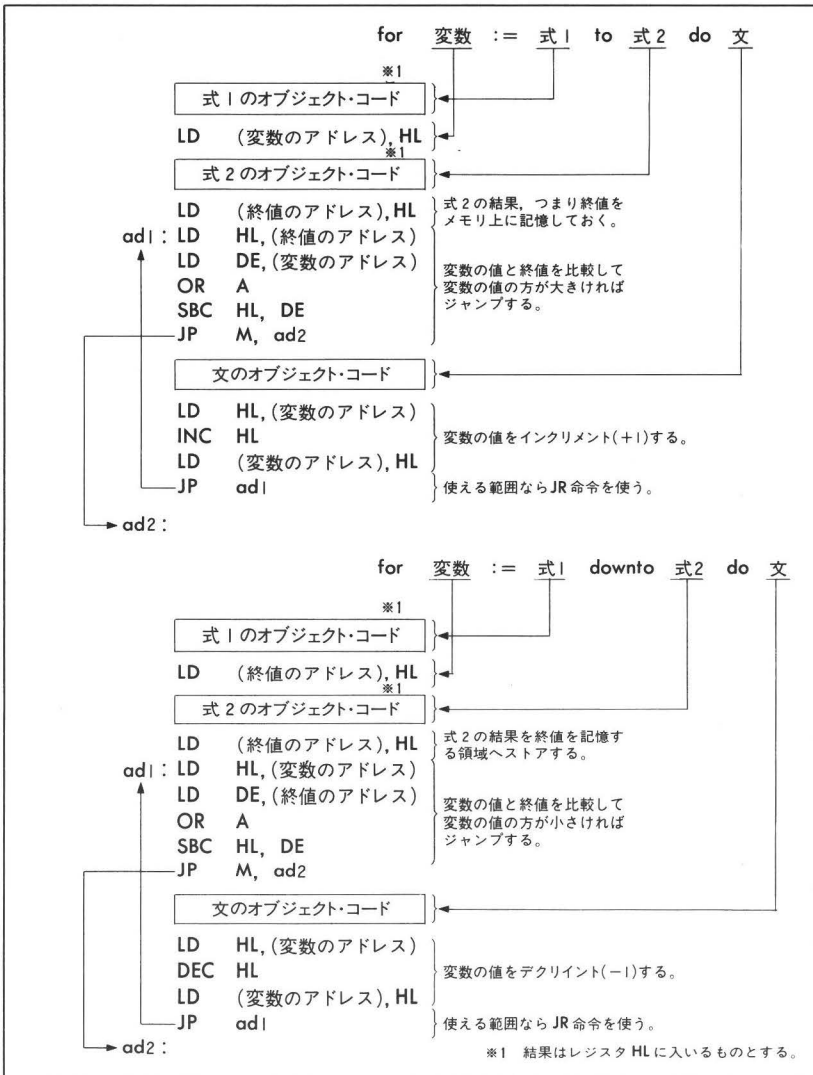
図3-45 FOR文をアセンブラ言語で表す

す。繰り返しのたびに計算していたのでは，終値の計算時間×繰り返し回数だけ遅くなってしまいます。繰り返しの前に計算してメモリに記憶しておけば，実行時間を速くし，しかも終値が入った変数の値をループ内で変えても，繰り返し回数に影響しません。ただし，終値が定数ならメモリ上に記憶するよりも制御変数と終値を示す定数とを直接比較するようにしたほうがよいでしょう。

図3-46 FOR文のオブジェクト・コード

図3-47 FOR文のコンパイル

②
現在のコード・ロケーション・カウンタの値を記憶しておく
lpad ← コード・ロケーション・カウンタ
dir＝0
?
No
Yes
変数の内容をレジスタHLへ，終値をレジスタDEへロードするコードを生成
"LD HL,(※1のアドレス)"
"LD DE,(※2のアドレス)"
のオブジェクト・コードを生成
終値をレジスタHLへ，変数の内容をレジスタDEへロードするコードを生成
"LD HL,(※2のアドレス)"
"LD DE,(※1のアドレス)"
のオブジェクト・コードを生成
レジスタHLとレジスタDEを比較するようなオブジェクト・コードを生成
"OR A"
"SBC HL, DE"
のオブジェクト・コードを生成
※3
現在のコード・ロケーション・カウンタの値を記憶しておく
stad
↑
コード・ロケーション・カウンタ
2の補数として HL＜DEならジャンプするようなオブジェクト・コードを生成
仮に
"JP M, 0000H"
というオブジェクト・コードを生成
次のトークンを読む
文
③
③
変数の内容をロードしてくるようなオブジェクト・コードを生成
"LD HL,(※1のアドレス)"
というオブジェクト・コードを生成
dir＝0
?
No
Yes
オブジェクト・コードとして "INC HL"を生成
オブジェクト・コードとして "DEC HL"を生成
変数へストアするようなオブジェクト・コードを生成
"LD (※1のアドレス), HL"というオブジェクト・コードを生成
ジャンプ命令を生成
"JP lpad" または
"JR lpad" のオブジェクト・コードを生成
仮に生成しておいた※3のジャンプ命令のアドレス部に正式のジャンプ先を設定するようなオブジェクト・コードを生成
jpad ← コード・ロケーション・カウンタ
コード・ロケーション・カウンタ ← stad＋1
"DEFW jpad" というオブジェクト・コードを生成
コード・ロケーション・カウンタ ← jpad
FOR文のコンパイルは終わり
注）この流れ図をプログラムにした場合，再帰的なプログラムになる。

Pascal の FOR 文にはありませんが，BASIC の FOR 文や FORTRAN の DO 文には増分の指定があります。Pascal では〝to〟と〝downto〟がそれぞれ増分として＋1，－1を示していますが，それ以外の増分は指定できません。BASIC の FOR 文や FORTRAN の DO 文は自由な増分が指定できます。たとえば BASIC の FOR 文（と NEXT 文）の構文は，**図3－48**の①のようになり，流れ図では②のようになります。アセンブラ言語では**図3－49**のようになります。増分を考えたときもコンパイル法は基本的には同じなので，**図3－47**の流れ図を一部変更すればコンパイルできます。

図3-48 FOR 文（BASIC）

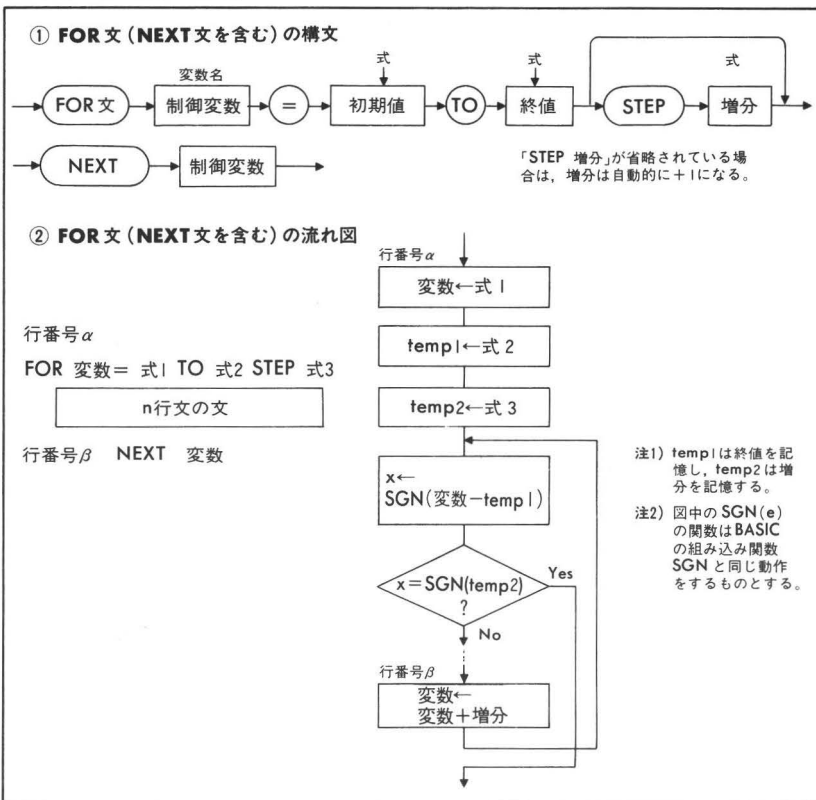
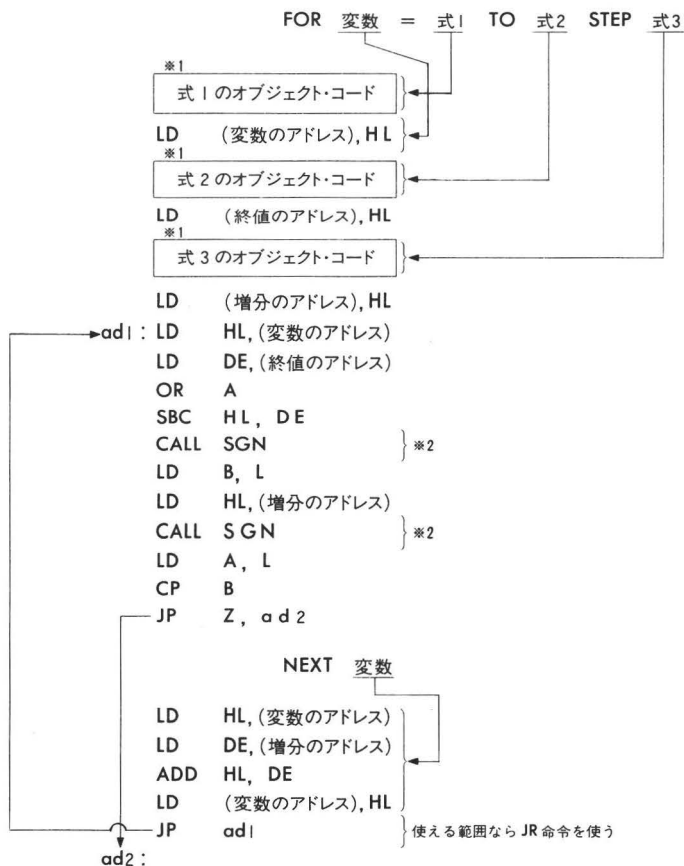



図3-49 FOR 文をアセンブラ言語で表す

注1) 終値と増分の計算結果はメモリ上に記憶される。それにより，FOR 文ごとにこの領域が取られる。

注2) 終値や増分が定数ならメモリ上には記憶せず，直接計算を行うようにする。

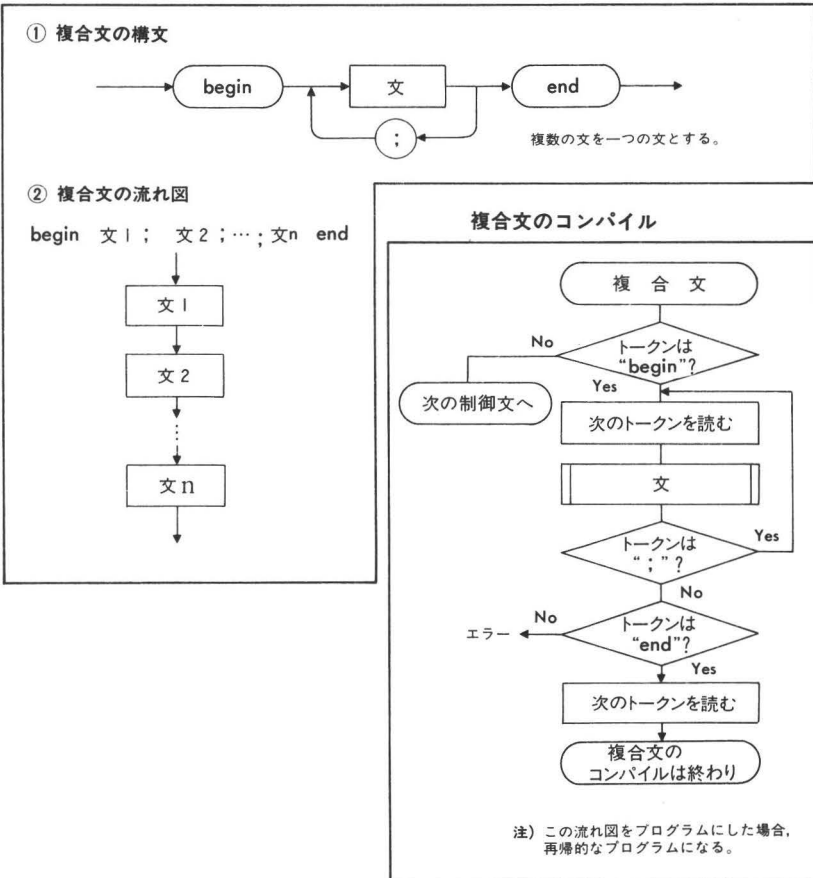
※1 結果はレジスタ HL に入いるものとする。

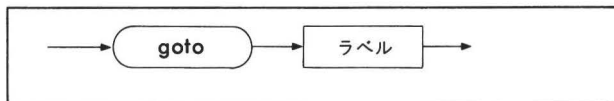
※2 SGN はランタイム・ルーチンで，レジスタ HLの符号を求める（レジスタ B は不変とする)。

HL > 0 ならば HL ← 0001H (+1)
HL = 0 ならば HL ← 0000H
HL < 0 ならば HL ← 0FFFFH (−1)

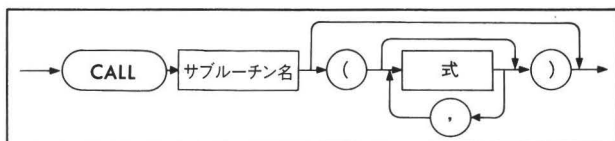
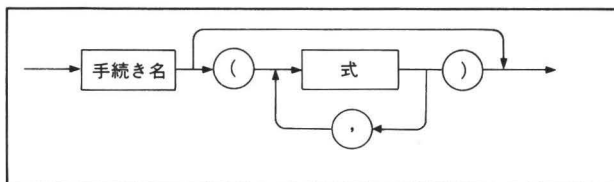
3-5-3 複合文

複合文は，一つの文しか書けない場所に複数の文を書きたいとき，まとめて一文とするものです。たとえば，IF 文の then や else の後，あるいは WHILE 文や FOR 文の do の後などに用いられます。構文は**図3－50**の①，流れ図では②のようになります。複合文は特定のオブジェクト・コードを生成しません。**図3－51**は複合文のコンパイル手順を流れ図にしたものです。

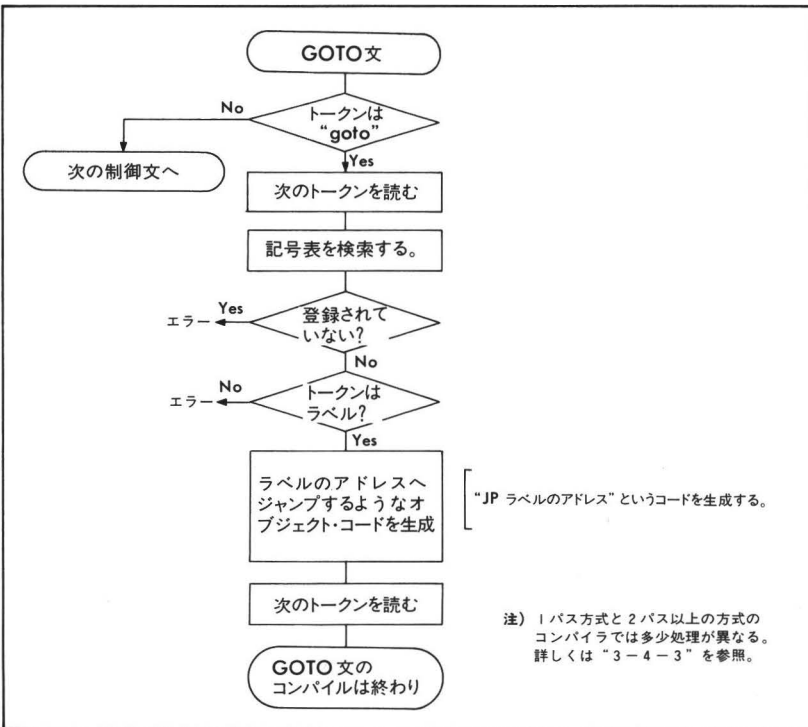
図3-50 複合文（Pascal）

3-5-4 GOTO文

GOTO 文は指定されたラベルへ実行を移す文で，構文は

となっています。コンパイラはGOTO 文を単純な無条件ジャンプの機械語に変換します。**図3－52**にGOTO 文のコンパイル手順を流れ図で示します。

3-5-5 サブルーチン(手続き)の呼び出し

サブルーチンの呼び出しの構文は，FORTRAN では

Pascal では

となっています。

FORTRAN は〝CALL〃というトークンを使うため，構文解析のとき〝CALL〃が見つかればサブルーチンの呼び出し処理に移れます。しかし，Pascal のように特定のトークンを用いず直接サブルーチン名が書かれるような場合は，名前だけでは代入文の左辺なのかサブルーチンの呼び出しなのかがわからないので，記号表を検索して識別します。また，Pascal はサブルーチンを呼び出す前に宣言をしなければならないという規則があるので記号表

図3-52 GOTO文のコンパイル

で識別できますが，そのような規則がない言語なら文を頭から解析していって代入文なのかサブルーチンの呼び出しなのかを識別しなければなりません（**3－4－3**参照)。その文がサブルーチンの呼び出しだとわかれば，後は関数の呼び出しと同じ処理でコンパイルすることができます（106ページ参照)。

さて，この項ではいくつかの制御文のオブジェクト・コードの生成例を載せましたが，これらの例は最適化をしていません。制御文の最適化はたいへん複雑で，他の文や式の最適化，コード生成などにも関係するため，パソコン用，特に8ビットCPUのコンパイラでは制御文の最適化をしていないものが多いようです。

3-6 オブジェクト・プログラムのメモリへの割りつけ

ここでは，コンパイラが生成したオブジェクト・コードや変数，配列などをメモリ・アドレスにどう割りつけるのか，その方法について述べていきます。

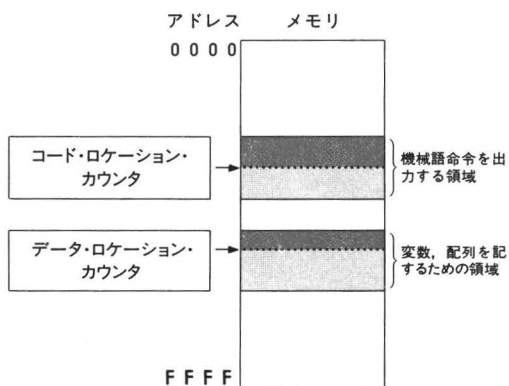
3-6-1 ロケーション・カウンタ

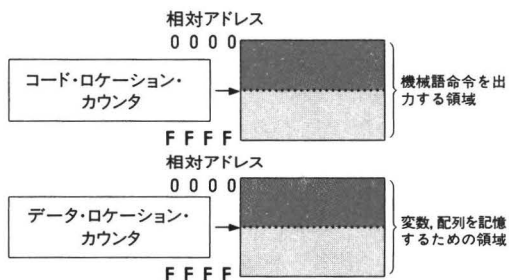
コンパイラには必ずロケーション・カウンタ（location counter）というものがあり，これが生成するオブジェクト・プログラムのアドレスを管理しています。ロケーション・カウンタには相対，あるいは絶対アドレスが記憶されていて，生成したオブジェクト・コードに対してアドレスを割りつけていきます。

コンパイラには，基本的に二つのロケーション・カウンタが必要で，一つはコンパイルし生成された機械語のアドレスを，もう一つは変数や配列などのアドレスを管理するものです。本書では，これらをそれぞれコード・ロケーション・カウンタ，データ・ロケーション・カウンタと呼んでいます。言語や出力形式によっては三つ以上のロケーション・カウンタを持つ場合もあります。三つ以上使う場合については後述するとして，基本的な二つのロケーション・カウンタがコンパイラ内でどのように使われているのか説明します。

ロケーション・カウンタに記憶するアドレスは**図3－53**のように，出力形式によってメモリ上の絶対アドレスか，メモリを意識しない相対アドレスかになります。アセンブラ言語を出力するコンパイラは文字列の操作だけでコンパイルできるため，ロケーション・カウンタを持たな

図3-53 ロケーション・カウンタとオブジェクト・プログラム

① 直接機械語を出力するコンパイラの場合

コード・ロケーション・カウンタやデータ・ロケーション・カウンタは，メモリ上の絶対アドレスで次に出力するオブジェクト・コードのアドレスや変数，配列を記憶するための領域のアドレスを覚えている。
この図はメモリ上に出力する場合を想定したものだが，ファイルへ出力する場合も考え方は同じ。

② 再配置可能なオブジェクト・プログラムを出力するコンパイラの場合

コード・ロケーション・カウンタやデータ・ロケーション・カウンタは，先頭をゼロとするような相対アドレスで次に出力するオブジェクト・コードのアドレスや変数，配列を記憶するための領域のアドレスを覚えている。コンパイラは，図のようなイメージでオブジェクト・プログラムを出力する。

（濃い網掛け）は実際にオブジェクト・コードが出力された部分や領域が取られた部分を示す。
………… は次にオブジェクト・コードを出力するアドレスや領域が取られるアドレスを示す。
（薄い網掛け）はこれからオブジェクト・コードが出力される部分や領域が取られる部分を示す。

いことがあります。

コード・ロケーション・カウンタはコードにアドレスを割りつけるために使われます。たとえばいま生成した××という1バイトのオブジェクト・コードは，現在のコード・ロケーション・カウンタが示すアドレスに割りつけられ，メモリまたはファイルに出力されます。そ

の後コード・ロケーション・カウンタは一つインクリメントされます。また，一時的にこのカウンタのアドレスを変更すれば任意のアドレスへオブジェクト・コードを出力させることもできます。**3－5**ではこの方法で未確定のジャンプ先アドレスを処理しています。

データ・ロケーション・カウンタはデータ，つまり変数や配列の領域を取るときに使われます。データに初期値がある場合は，データにアドレスが割りつけられ，初期値も出力されます。初期値がないときはアドレスだけが割りつけられ出力はしません。たとえば，初期値ｘの変数Ｖ１（ｎバイト）の領域には現在のデータ・ロケーション・カウンタのアドレスが割りつけられ，初期値ｘはそのアドレスでメモリやファイルに出力されます。そしてデータ・ロケーション・カウンタは変数の大きさ（ｎバイト）分インクリメントされます。また，初期値が必要ない変数Ｖ２（ｍバイト）のときは，アドレスが割りつけられてからｍバイト分インクリメントされるだけです。

さて，三つ目以上のロケーション・カウンタですが，これも変数や配列を管理するために持つことが多いようです。

たとえば，FORTRANの変数や配列には局所的なものと，プログラム中で共通に使われるコモン・ブロック内のものとがあります。このような場合，これらを同列に扱うことばできないのでデータ・ロケーション・カウンタ以外にコモン・ブロック用のロケーション・カウンタが必要になります。

また，変数，配列に対して固定アドレス，つまり静的なアドレスを割りつけるだけでなく，言語によっては局所的な変数，配列の全部または一部をスタック上に動的に取ることがあります。そのときにも静的，動的，二つのデータ・ロケーション・カウンタが必要になります。

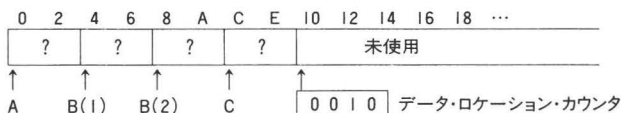
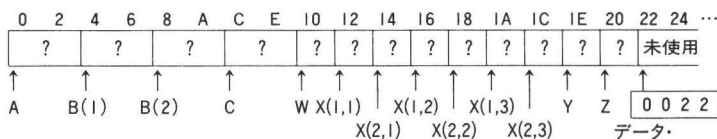
3-6-2　宣言文とメモリの割りつけ

ここでは，宣言文によって宣言された変数や配列に対するアドレスの割りつけについて説明します。

〔1〕静的なアドレスの割りつけ

BASICやFORTRANは，変数や配列に対して固定されたアドレス，つまり静的なアドレスを割りつけます。また，静的なアドレスが割りつけられた変数や配列には，コンパイル時に初期値の設定ができます。

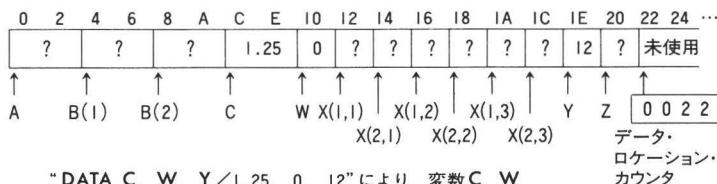
たとえばFORTRANで

```
REAL  A, B (2), C
INTEGER  W, X (2, 3), Y, Z
DATA  C, W, Y/1.25, 0, 12/
```

という文があって実数4バイト，整数2バイトの領域が取られるならば，コンパイラはメモリ上に図3－54のように各変数，配列にアドレスを割りつけ，初期値を設定します。

静的なアドレスが割りつけられた変数や配列が式の中で使われると，その要素のロード，ストアは絶対アドレスに対して行われます（これまでの説明で使ってきた変数，配列要素のロード，ストアはすべてこの方法)。

〔2〕動的なアドレスの割りつけ

Pascalのように再帰的なプログラムが許される言語では，局所的な変数，配列を実行時にスタック上に動的に取るため，サブルーチンや関数ごとに，局所的変数（配列）のアドレスが異なってきます。そこで，コンパイル時にはサブルーチンや関数ごとに，局所的変数（配列）へのロケーション・カウンタを用意して相対的なアドレスを割りつけるようにします。この場合はコンパイル時の初期値設定ができません。どうしても設定したいとき

図3-54 静的なアドレスの割り付け（FORTRAN）

は初期値をストアするようなオブジェクト・コード（命令）を生成するしかありません。

動的なアドレスが割りつけられた変数や配列は，静的なアドレスが割りつけられたときとはちがい，コンパイル終了時にはメモリ上の大きさがわかりません。

実行中，局所的変数をどのようにスタック上に取るかを図3－55，図3－56に示します。この図はPascalのようなブロック構造の言語を再帰的に実行するときを考えて

図3-55　実行中の１ブロック内の局所的な変数や配列の様子

　再帰的に呼び出されるサブルーチン(手続き)や関数は，ルーチンの始めで右図のようにスタック上に局所的な変数，配列の領域を取る。この領域は実行が終わり，元のルーチンに戻るとき解放する。

　右図の中のＢＰとはベース・ポインタのことで，このポインタが示す絶対アドレスを基準に局所的な変数や配列のロードとストアが行われる。ＢＰにはＣＰＵのインデックス・レジスタなどが使われる。ＳＰはスタック・ポインタのこと。

　スタック上に取られるブロック一つ分のデータの内容は次のとおり。

●BPの元の値

　元のルーチンに戻ったときに，BPの値も元の値に戻さなければならないので記憶しておく。

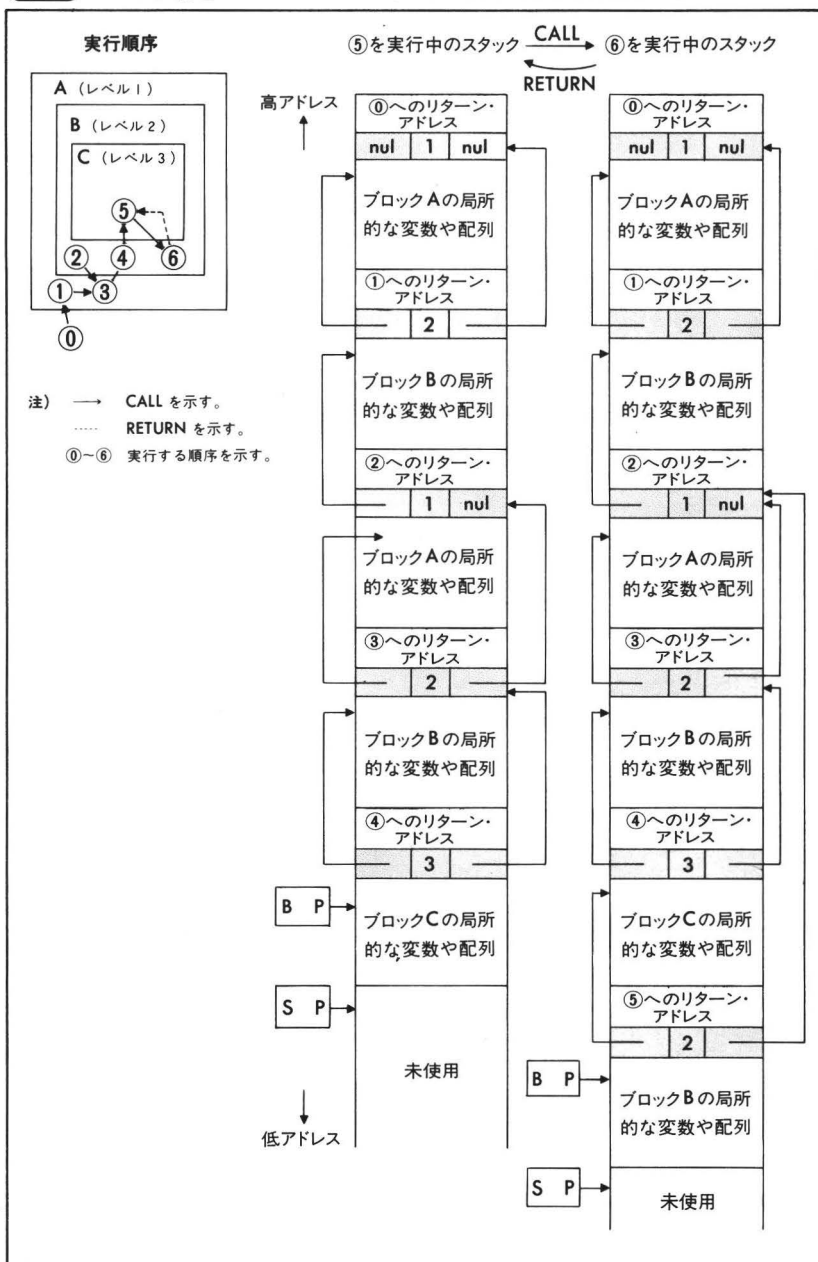
●ブロックのレベル番号

　以前に名前には有効範囲があると説明したが，このレベル番号によって名前（この場合変数や配列の有効範囲）を決める。

●レベル番号が一つの下のブロックへのポインタ

　BPが示すブロック外の変数，配列に対してロード，ストアを行うとき，このポインタをたどって行き目的の変数，配列をアクセスする。

います。この場合，単にスタック上に領域を取るだけでなく，どのように実行されてきたかの履歴も保存しなければなりません。それは，ブロック構造では名前に有効範囲があるからです。たとえば，**図3－56**でブロックＢの②を実行しているときのブロックＡの変数ａの値と，ブロックＢの⑥を実行しているときのブロックＡの変数ａの値はまったく異なります。このような場合でも，ブロ

図3-56 ブロック構造のプログラムの再帰的な実行とスタック上のデータ

ック外の変数や配列を正しくアクセスするために，スタック上に一つ下のブロックへのポインタがあるのです。このポインタは，局所的な変数，配列がスタックに取られるときに一つ下のレベルのブロックを探して示すように設定します。

動的なアドレスが割りつけられた変数や配列のアクセスは，間接アドレスによるしかありません。間接アドレスの扱いが不得意なCPUでは既存の命令の組み合わせでアクセスしなければならず，オブジェクト・コードの増大を招きます。こんなときは，間接アドレスでのアクセスを一つのサブルーチンとしてランタイム・ルーチンをつくります。

変数や配列が動的なアドレスで割りつけられたプログラムは，静的に割りつけられたプログラムに比べ，実行時間が遅くなります。ですから，全域的な（レベル0ブロックの）変数や配列は静的なアドレスを割りつけるようにして，少しでも実行時間を速くします。

3-7 サブルーチンと関数の処理

この項ではサブルーチンや関数への引数の渡しかたや渡された引数の処理，また，式の中に使われた引数の扱いかた，およびサブルーチンや関数のコンパイル手順について説明します。

3-7-1 引数の渡しかたと引数の処理

引数の渡しかたには，引数の値を記憶したアドレスを渡す場合と，値だけを渡す場合の二通りがあります。言語によってはこのどちらかしか扱えないものや，サブルーチン，関数の宣言のときに指定できるものなどがあります。

〔1〕引数としてアドレスが渡される場合

引数としてアドレスを渡す言語の一つにFORTRANがあります。サブルーチンの呼び出し側に

```
CALL SUB1 (A,I, 36, K+5)
```

という文があったとすると，サブルーチンSUB1には変数AとI，定数36，そして式K＋5の演算結果の記憶場所のアドレスを渡すことになります。そのアドレスはコンパイル時に，定数としてメモリに記憶し，サブルーチンを呼び出すときには，そのメモリのアドレスを渡すようにします。そして呼び出されるサブルーチンの側では，たとえばSUB1が

```
SUBROUTINE SUB1 (B, L, M, N)
    ∫
  END
```

であったら，まず渡された実引数のアドレスを仮引数B，L，M，Nにコピーします。仮引数は変数そのものの値でなくアドレスを記憶しているため，仮引数が式の中で使われたときは間接アドレスによってアクセスしなければなりません(**図3-57**)。**図3-58**が引数として配列全体を渡す場合，**図3-59**がサブルーチン内で他のサブルーチンを呼ぶ場合に仮引数を実引数として渡す場合です。

ここではFORTRANを例としているので，再帰的なプログラムをつくらないことを前提にしています。なぜ再帰的なプログラムがだめなのかというと，実引数のアドレス，および式の演算結果や実数などを記憶するワーク・エリアをコンパイル時に静的に（メモリ上に）設定しているからです。再帰的な呼び出しを許すような言語では，これらを実行時にスタックに設定するようにしています。そして，サブルーチン側でも仮引数の領域をスタックに取るようにします。このようにすべての領域をスタックに取れば再帰的な呼び出しが可能になります。

図3-57 引数としてアドレスを渡す場合（FORTRAN）

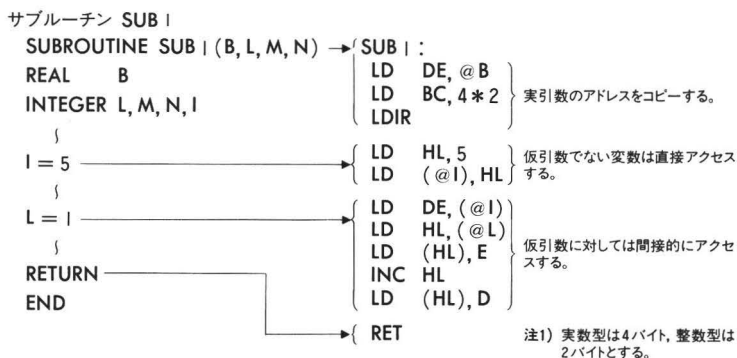
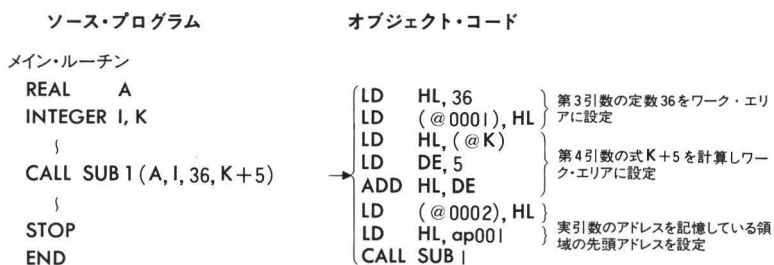
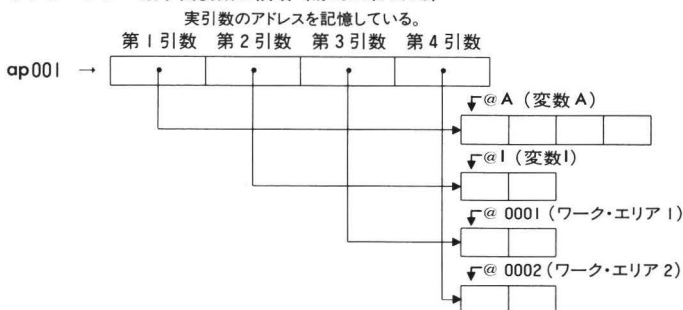



図3-58 引数としてアドレスを渡す，配列の場合（FORTRAN）

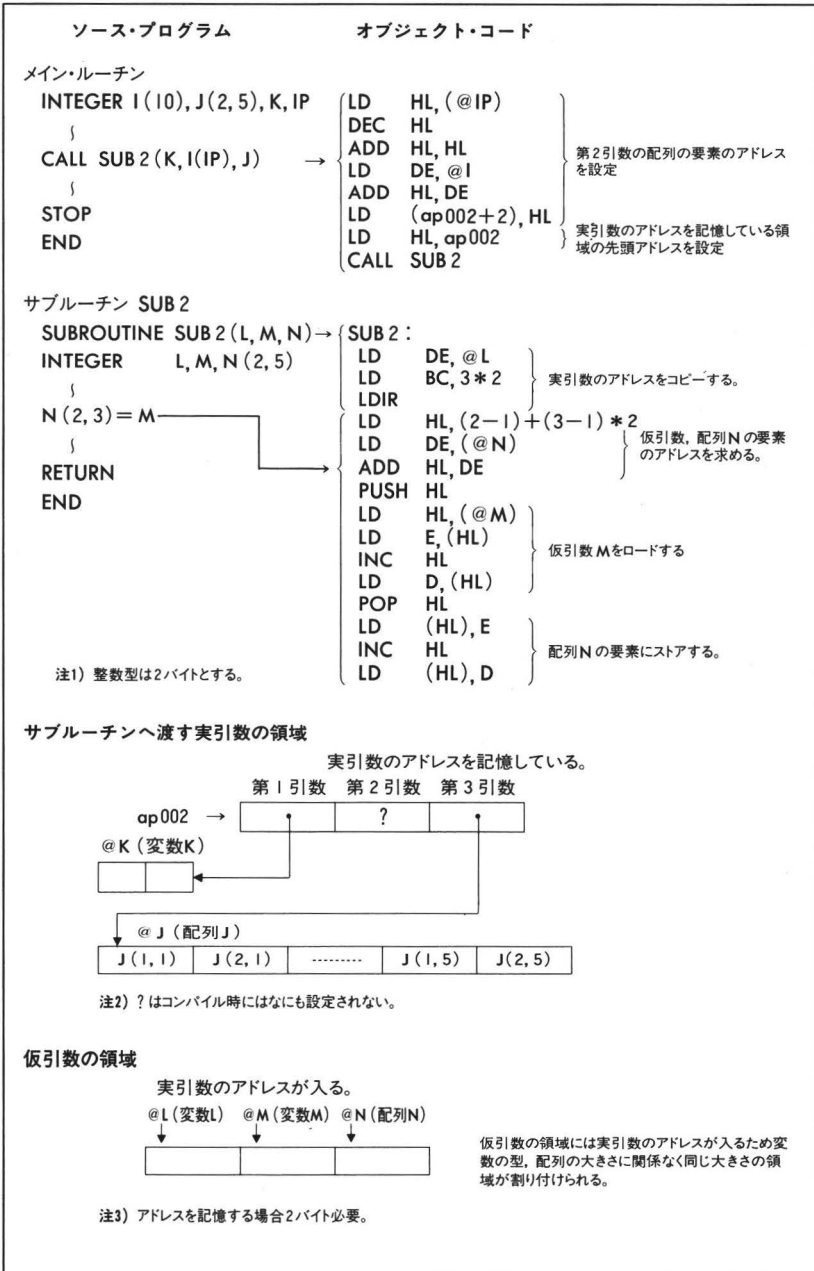
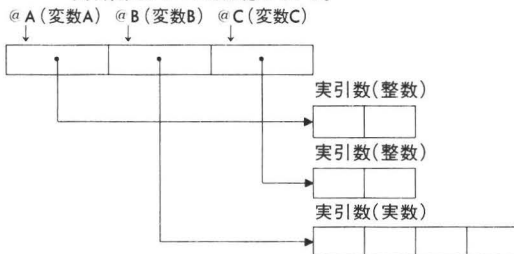
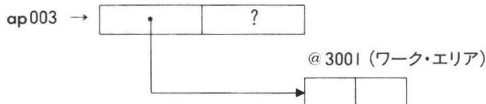



図3-59 引数としてアドレスを渡す，仮引数の場合（FORTRAN）

〔2〕引数として値が渡される場合

引数として値を渡す言語にはPascalがあります。Pascalではアドレスを渡すことも可能（仮引数に“var”を宣言する）ですが，ここでは値がどのようにサブルーチンに渡されるのかを見てみます。

Pascalでは再帰的なプログラムが許されているので，局所的変数や配列はすべてスタック上に取られています。実引数も例外ではなくスタック上に設定しサブルーチンに渡します。たとえば

```
proc1 (a,b,36,d+5)
```

という文で変数aが実数(4バイト長)，変数bとdが整数（2バイト長）だったとしますと，d＋5の演算結果，定数36，変数bの値，変数aの値の順にスタックへPUSHします。**図3-60**の①が四つの実引数の値をPUSHし終わったスタックの状態です。そして呼び出されたサブルーチン側では，渡された実引数の各値を局所的な変数と同じように扱います。つまり，仮引数へはベース・ポインタによる間接アドレスでアクセスするわけです。**図3-60**の②がサブルーチン実行中のスタックの状態です。

こんどは値の一つ一つではなく，配列全体の値を渡す場合です。基本的には変数値をスタックに入れるのと同じことで，配列の全要素をスタックへ入れます。**図3-61**は配列全体の値を渡す場合の例です。

引数の値を渡す言語のコンパイラでは，サブルーチンから呼び出し側に実行結果を渡す目的で仮引数を使うことができないので，このような値は全域的な変数や配列で渡すようにします。

図3-60 引数として値を渡す場合(アドレスを渡す機能がないサブセットのPascal)

実行時のスタックの状態
高アドレス
+17
仮引数 z（d＋5の値）
+15
仮引数 y（定数36）
+13
仮引数 x（変数bの値）
仮引数 w（変数aの値）
+9
呼び出し側で設定した実引数の値
+7
リターン・アドレス
CALL命令によってCPUが自動的に入れたリターン・アドレス
+5
元のレジスタIX の内容
+3
現ブロックのレベル番号
+1
一つレベルが下のブロックへのポインタ
IX →
−1
変数 p
−3
変数 q
−5
変数 r
−7
変数 s
−9
変数 t
ランタイム・ルーチンLALLOCで取られた領域，ランタイム・ルーチンLFREEを実行するとこの領域は解放される。
SP →
未使用
注）インデックス・レジスタIXをベース・ポインタに使っている。
低アドレス

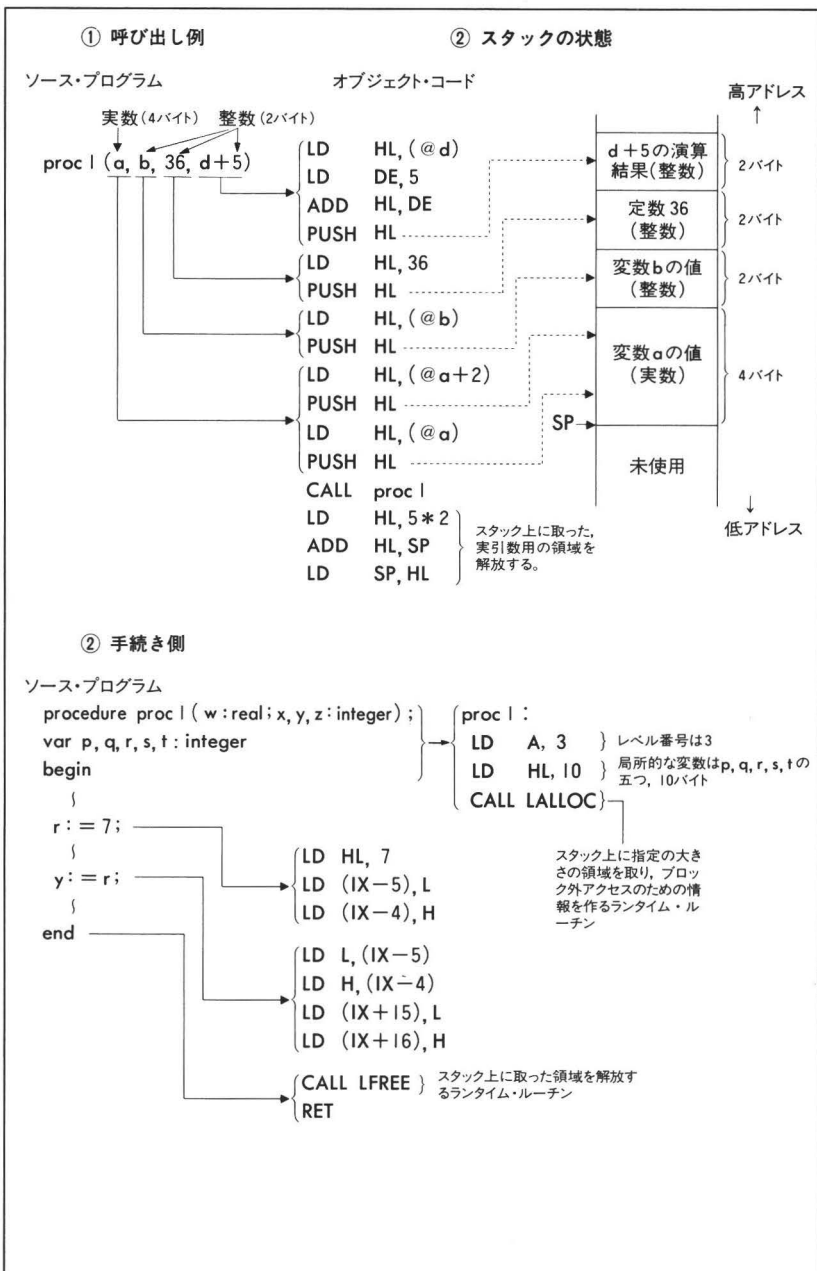
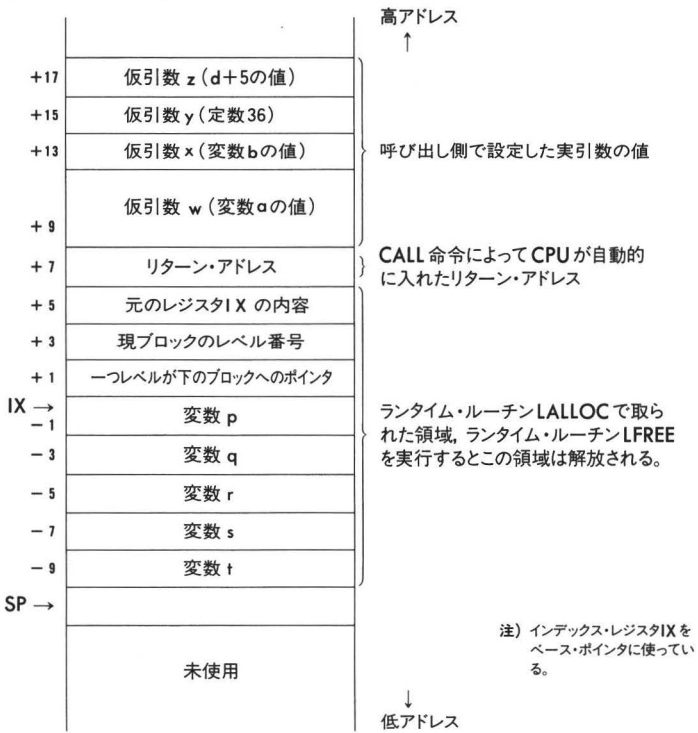



図3-61 引数として値を渡す,配列の場合(アドレスを渡す機能がないサブセットのPascal)

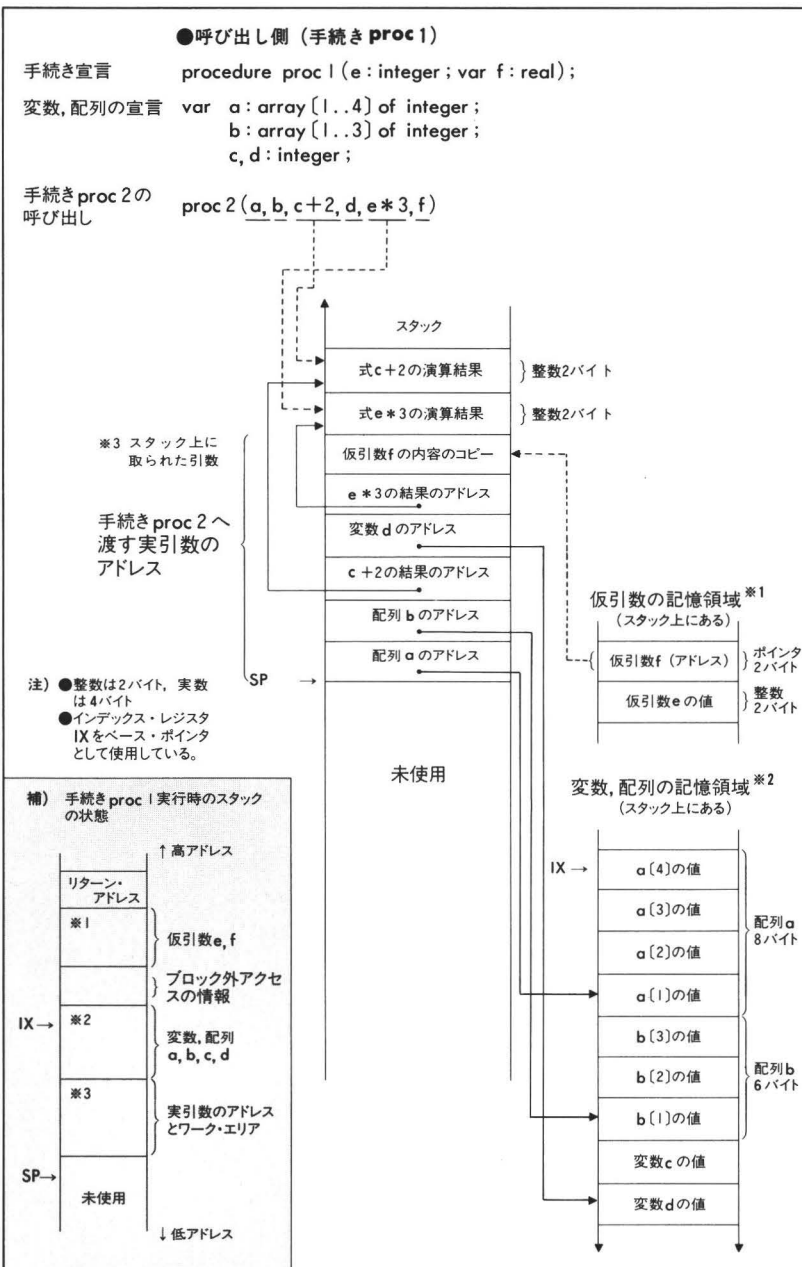
〔3〕サブルーチンや関数の宣言のとき，渡す内容を指定できる場合

本来，Pascal は引数として値，アドレスの両方を扱うことができます。まず呼び出し側は，基本的には〔1〕で説明した方法で実引数のアドレスをサブルーチンへ渡します。呼ばれるサブルーチン側では，実引数の値がほしいのなら渡されたアドレスにある値を仮引数へ転送し，アドレスがほしいのなら渡されたアドレスをそのままコピーします。サブルーチン内では，仮引数に値が渡されているなら局所的な変数，配列と同じようにアクセスし，アドレスが渡されているならそのアドレスから間接的にアクセスします。

Pascal では再帰的な呼び出しを許すため，このような操作はすべてスタック上で行います。**図3-62**はスタックへの引数の設定のしかた，**図3-63**はこのような引数の渡しかたをするオブジェクト・コードです。

関数の引数についてもまったく同様です。ただ関数は式の中で呼ばれるため，関数実行後に値を戻さなければならず，そのための処理が必要になります。これは引数には関係ないのですが，局所的な変数や配列を取る処理，呼ばれたルーチンへ戻る処理などに関係してきます。

最後に実引数と仮引数の個数のチェックです。引数の個数のチェック方法には二つあり，一つはコンパイル中にチェックする方法，もう一つはチェックのためのオブジェクト・コードを生成し実行中にチェックする方法です。実行速度とオブジェクト・コードの大きさの点から，できれば実行中のチェックは避けた方がよいでしょう。パソコン用のコンパイラでは引数の個数をチェックしないものが多いようですが，やはりこの処理は必要だと思います。

図3-62 引数としてアドレス，値の両方が渡せる場合(標準 Pascal)

●手続き proc 2 側

手続き宣言

```
procedure proc 2
    (g : array〔1..4〕of integer ;
     var h : array〔1..3〕of integer ;
     i : integer ;
     var j : integer ;
     k : integer ;
     l : real )
```

変数の宣言

```
var x, y : integer ;
```

手続き proc 2 実行時のスタックの状態

区分	番地	内容	転送
渡された実引数のアドレス		仮引数f のコピー	アドレスが示す内容を転送
		e ＊3の結果のアドレス	アドレスが示す内容を転送
		変数 dのアドレス	そのままコピーする。
		c ＋1 の結果のアドレス	アドレスが示す内容を転送
		配列 bのアドレス	そのままコピーする。
		配列 aのアドレス	アドレスが示す配列要素をすべて転送
	+27	リターン・アドレス	
仮引数の領域	+23	仮引数 l （値）	
	+21	仮引数K （値）	
	+19	仮引数 j （アドレス）	
	+17	仮引数i （値）	
	+15	仮引数h （アドレス）	
	+13	仮引数g〔4〕（値）	
	+11	仮引数g〔3〕（値）	
	+9	仮引数g〔2〕（値）	
	+7	仮引数g〔1〕（値）	
ブロック外アクセスの情報	+5	元のレジスタIXの内容	
	+3	現ブロックのレベル番号	
	+1	一つレベルが下のブロックへのポインタ	
	IX → −1	変数 x	
	−3	変数 y	
	SP →	未使用	

↓ 低アドレス

図3-63 引数としてアドレス, 値の両方が渡せる場合のオブジェクト・コード（標準 Pascal）

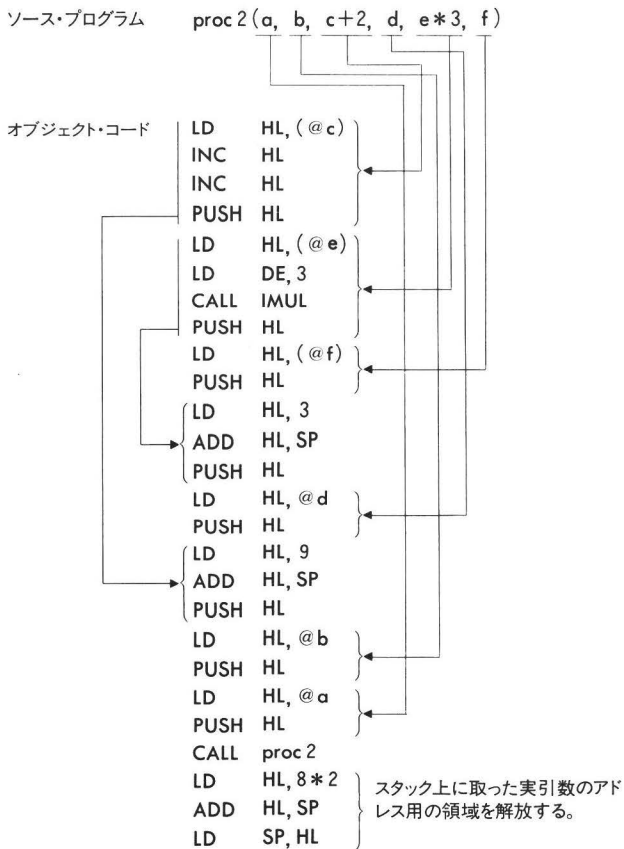
図3-62のようなスタック状態になるオブジェクト・コード。

●呼び出し側（手続き proc 1）

ソース・プログラム　proc 2 (a, b, c+2, d, e*3, f)

オブジェクト・コード

```
LD    HL, (@c)
INC   HL
INC   HL
PUSH  HL
LD    HL, (@e)
LD    DE, 3
CALL  IMUL
PUSH  HL
LD    HL, (@f)
PUSH  HL
LD    HL, 3
ADD   HL, SP
PUSH  HL
LD    HL, @d
PUSH  HL
LD    HL, 9
ADD   HL, SP
PUSH  HL
LD    HL, @b
PUSH  HL
LD    HL, @a
PUSH  HL
CALL  proc 2
LD    HL, 8*2
ADD   HL, SP
LD    SP, HL
```

最後の3命令（LD HL, 8*2 / ADD HL, SP / LD SP, HL）：スタック上に取った実引数のアドレス用の領域を解放する。

●手続き proc 2 側

ソース・プログラム

```
procedure proc 2 (g : array〔1..4〕of integer ;
    var h : array〔1..3〕of integer ; i : integer ;
    var j : integer ; k : integer ; l : real)
var x, y : integer ;
begin ;
    ∫
end ;
```

オブジェクト・コード

```
proc 2 :
    LD      HL, 7*2     } 実引数のアドレスを記憶している領域
    ADD     HL, SP      } の最終アドレスを求める。
    LD      A, 6        } 仮引数の数は6
    CALL    SETDAG      } スタック上に仮引数の領域を取りそこ
                          へ，実引数の値またはアドレスを入れ
                          るランタイム・ルーチン
    DEFW    4           → 第6引数，値4バイト
    DEFW    2           → 第5引数，値2バイト
    DEFW    0           → 第4引数，アドレス     } 仮引数の
    DEFW    2           → 第3引数，値2バイト   } 内容を示す
    DEFW    0           → 第2引数，アドレス
    DEFW    4*2         → 第1引数，値8バイト
    LD      A, 3        } レベル番号は3
    LD      HL, 4       } 局所的な変数はx, yの二つ，4バイト
    CALL    LALLOC      } スタック上に指定の大きさの領域を取
                          り，ブロック外アクセスのための情報
                          を作るランタイム・ルーチン
      ∫
    CALL    LFREE       } ランタイム・ルーチンLALLOCで取っ
                          たスタック上の領域を解放するランタ
                          イム・ルーチン
    LD      HL, 20      }
    ADD     HL, SP      } スタック上にとった仮引数の領域を解
    LD      SP, HL      } 放する。
    RET
```

3-7-2 サブルーチンと関数のコンパイル

ここでは，サブルーチンや関数のコンパイルをFORTRANとPascalを例にして説明します。

サブルーチンや関数は**図3-64**のように三つの部分に分けられます。まずコンパイラに対してサブルーチン名や関数名，引数，そのルーチン内で使う変数や配列の宣言を行う部分です。次に実際の処理を行う部分，そして最後が終わりを表す部分です。サブルーチンや関数はこの順序で処理されていきます。

〔1〕サブルーチン，関数の始まりの識別と処理

コンパイラはサブルーチンや関数の始まりを文や宣言によって識別します。これらの文や宣言にはサブルーチンなら名前が，関数なら名前と型，実引数を入れるための仮引数が書かれています。コンパイラは記号表へサブルーチン名（関数名）とその型を登録し，仮引数には実引数のアドレスまたは値を記憶する領域を割りつけ，仮引数の名前を記号表へ登録します。そして，実引数を受け取るためのコードを生成します。

サブルーチンや関数内で宣言された変数や配列は局所的な名前として記憶領域が割りつけられ，記号表へ登録します。

FORTRANでは，仮引数や変数，配列は実行時にメモリ上に静的に取られます。Pascalはブロック構造を持った再帰的呼び出し可能な言語なので，仮引数や変数，配列は実行時にスタック上に動的に取られます。つまりコンパイル時にはベース・レジスタを中心とする相対アドレスを割りつけ，実行時にその領域をスタックに取るようなオブジェクト・コードを生成してやります。また，Pascalは変数などの宣言の後にそのルーチン内だけで使うサブルーチンや関数が宣言できるため，コンパイラはサブルーチン，関数の処理を再帰的に行うようにしま

図3-64 サブルーチンや関数の構成

① サブルーチン（手続き）の構成

● FORTRAN の場合

構成	説明
SUBROUTINE サブルーチン名（引数,…） 変数や配列の型などの宣言	サブルーチンの始まりを示す。コンパイラに対してサブルーチン名，引数，変数，配列などを宣言する部分。
実行文	実際に処理を行う部分。
END	END 文によりサブルーチンの終わりを示す。

● Pascal の場合

構成	説明
procedure 手続き名（引数,……）； 変数や配列の宣言と手続き，関数の宣言	手続きの始まりを示す。コンパイラに対して手続き名，変数，配列などを宣言するのとこの手続きのみで使用する手続き，関数を宣言する。
begin 実行文	実際に処理を行う部分。"begin" により始まる。
end；	"end" により手続きの終わりを示す。

② 関数の構成

● FORTRAN の場合

構成	説明
型 FUNCTION 関数名（引数，…） 変数や配列の型などの宣言	関数の始まりを示す。コンパイラに対して関数名と型，引数，変数，配列などを宣言する部分。
実行文	実際に処理を行なう部分。
END	END 文により関数の終わりを示す。

● Pascal の場合

構成	説明
function 関数名（引数，…）：型； 変数や配列などの宣言と手続き，関数の宣言	関数の始まりを示す。コンパイラに対して関数名と型，引数，変数，配列などを宣言するのとこの関数のみで使用する手続き，関数を宣言する。
begin 実行文	実際に処理を行う部分。"begin" により始まる。
end；	"end" により関数の終わりを示す。

す。もし，宣言されていた場合はいまコンパイル中のサブルーチンや関数の処理を中止し，先に宣言されたサブルーチンや関数をコンパイルします。

〔2〕サブルーチン，関数の処理本体

この部分に書かれる文は代入文や制御文，入出力文などです。代入文や制御文のコンパイルは前述の通り，入出力文のコンパイルは主として入出力ランタイム・ルーチンを呼び出すオブジェクト・コードを生成することで，構文は多少異なってもサブルーチンの呼び出しと同じです。ここで注意しなければならないのは，式の中の変数，配列の処理です。式の中で使われる変数や配列は型や属性(引数なのか，局所的な変数なのか)，どのブロックで宣言されたものか，などのことがわからないからです。そこでコンパイラは，変数や配列の要素をアクセスするようなオブジェクト・コードを生成するときには記号表を検索し，その変数の属性がどのようになっているのか確かめる必要があります。

〔3〕サブルーチン，関数の終わり

FORTRAN は END 文によりサブルーチンや関数の終わりを知り，実行文のコンパイルを終わり，次のサブルーチンや関数のコンパイルの準備をします。

Pascal ではサブルーチンや関数の処理本体は一つの複合文なので，複合文の終わりがサブルーチンや関数の終わりとなります。Pascal には FORTRAN の RETURN 文のような文がないので，サブルーチンや関数の終わりでスタック上の変数領域を解放し，呼んだルーチンへ戻るようなオブジェクト・コードを生成します。そして，コンパイラはいまコンパイルが終わったサブルーチンや関数を宣言していたサブルーチンあるいは関数のコンパイルを再開します。

以上,〔1〕~〔3〕までサブルーチンや関数のコンパイル方法を述べましたが,メイン・ルーチンも始まりの識別が多少異なることと次の点を除き,まったく同じです。それは,プログラムは必ずメイン・ルーチンから実行されるということです。また,FORTRAN のようにメイン・ルーチン内では書いてはいけない文がある言語では,そのような文が使われたときはエラーとするような処理が必要です。

最後にメイン・ルーチンも含めたサブルーチン,関数のコンパイルを流れ図にしたものを**図3-65**に示します。

図3-65 サブルーチン,関数のコンパイル(メイン・ルーチンを含む)

①
END 文
Yes
No
次の行を読む
EOF
制御文？
Yes
制御文の
コンパイル
次のサブルーチン、
関数のコンパイルの
準備
コンパイル終わり
No
入出力文？
Yes
入出力文の
コンパイル
②
No
代入文？
Yes
代入文の
コンパイル
No
その他の文の処理
（FORMAT文など）
次の行を読む
EOF
エラー
注）注釈行は入力されないものとしている。

② **Pascal の場合**

コンパイル
コンパイルの準備
始めのトークンを読む
エラー
No
トークンは
"program"?
Yes
プログラム名，カッコ
（ ）内のファイル名を
読み込む
トークン"program"の次は
プログラム名（ファイル名，…）
となっている。
エラー
No
")"の次
の文字は";"?
Yes
次のトークンを読む
ブロックのコンパイル
エラー
No
"."でプログラム
が終わっているか?
Yes
コンパイル終わり

ブロックのコンパイル
"label"?
Yes
ラベルの宣言
No
"const"?
Yes
定数の定義
No
"type"?
Yes
型の定義
No
"var"?
Yes
変数,配列の宣言
No
"procedure"?
Yes
手続きの始まり。手続き名を記号表へ登録
引数があれば,引数にアドレスを割り付け記号表へ登録
No
"function"?
Yes
関数の始まり。手続き名を記号表へ登録
引数があれば,引数にアドレスを割り付け記号表へ登録
関数の型が指定されているのでその処理
No
"begin"?
Yes
文のコンパイル
制御文や代入文
No
エラー
";"?
Yes
No
"end"?
No
エラー
Yes
リターン
";"?
No
エラー
Yes
ブロックのコンパイル
ブロックの終わりは";"?
Yes
No
エラー

3-8 コンパイラ全体の構成

コンパイラの設計手法も前項までで一通り説明し終わったので，最後にコンパイラ全体についてもう一度考えてみることにします。

図3-66は1パス方式のコンパイラの起動から終了までの制御の流れとデータの流れです。コンパイラが起動されると，まずイニシャライズ・ルーチンでコンパイラのワーク・エリアや記号表をイニシャライズ（初期化）します。このときイニシャライズ・ルーチンは起動時に

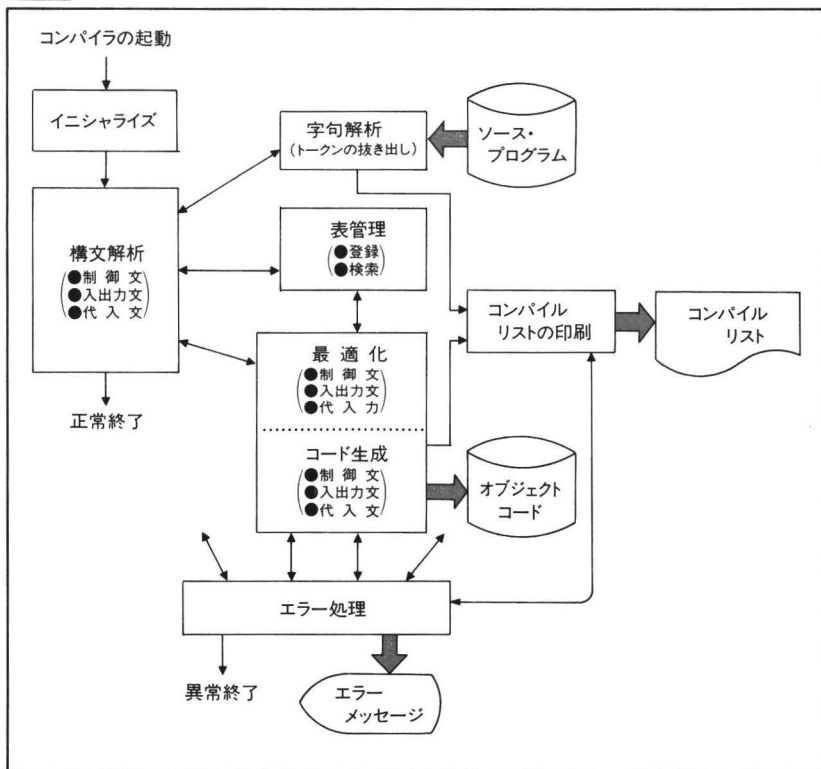
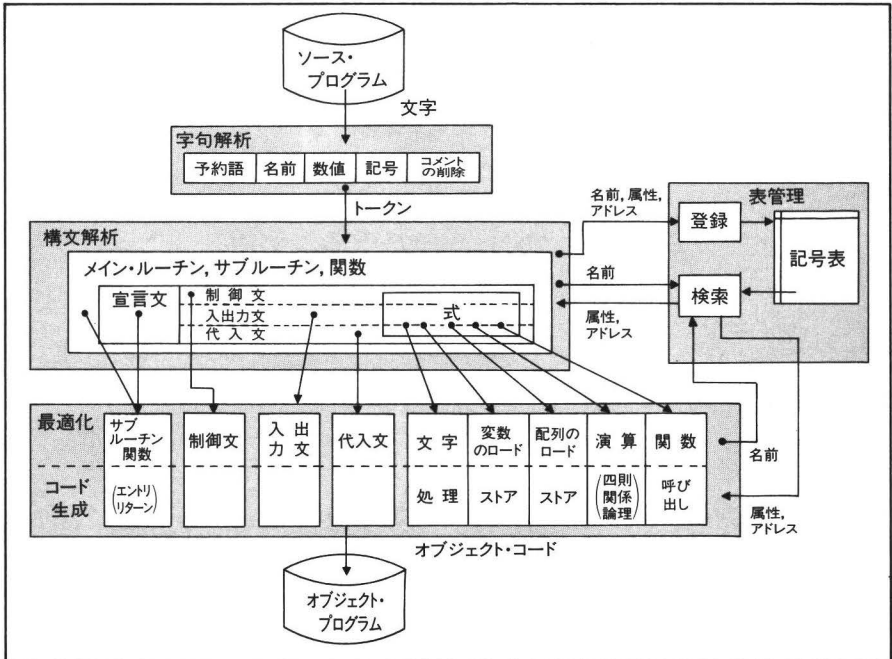
図3-66 コンパイラの構成と制御の流れ（1パス方式の場合）

指定されるソース・プログラム，オブジェクト・プログラムのファイルをオープンし，スイッチやオプションを解析し，コンパイルの準備をします。イニシャライズが終わると構文解析へ行き，コンパイルを始めます。コンパイル中は構文解析が常に処理の中心にいて，字句解析，最適化，コード生成は構文解析の下で仕事をします。コンパイルの終了も構文解析が判断します。コンパイル終了時にはオープンされているすべてのファイルをクローズし，ワーク・ファイルをすべて削除します。**図3-67**はコンパイルしているときのデータの流れと処理をある程度詳しく表した図です。矢印がデータの流れを，四角が処理を表します。字句解析ではソース・プログラムから文字を入力し，予約語，名前，数値などトークンがつくられます。構文解析ではトークンを入力し，文法にしたがって構文を解析し，その結果により次の構文の処理へ

図3-67 コンパイラ内の処理構造

行ったり，最適化，コード生成ヘオブジェクト・コードの生成を依頼します。最適化，コード生成は構文解析から依頼されるすべてのオブジェクト・コードが生成できるよう，いくつもの生成処理を持つことになります。

このように，ソース・プログラムはいくつもの処理を通ってオブジェクト・プログラムとなります。各処理にはエラーのチェックがあり，誤ったオブジェクト・プログラムがつくられないようにしています。もし，エラーを発見した場合はエラー処理へ飛ぶようになっています。エラー処理ではエラーメッセージを表示し，エラー回復が可能ならエラー回復の処理をしてコンパイルを続行します。エラー回復が不可能なときは，異常終了ということでコンパイル終了の処理に移ります。

またコンパイラには，ソース・プログラム，生成したオブジェクト・コード，エラーメッセージ，オブジェクト・プログラムの大きさやアドレスなどをコンパイル・リストとして印刷する処理もあります。

このような処理が集まってコンパイラがつくられています。ただし，ここで示したコンパイラの構成はパスの回数や言語の内容によって変わってきます。

最後に個人レベルで小さなコンパイラを設計，作成するときのアドバイスをまとめておきます。.

●構文解析は再帰的な方法のほうが簡単です。

●最適化は式の演算順序を入れ換える程度でよいと思います。あまり複雑な最適化は設計も大変で，コンパイラ自体も大きくなるからです。

●エラー処理は，後のコンパイルに影響を与えるエラーなら異常終了させるようにします。それは，そのようなエラーを回復させるのが大変な処理だからです。それに，不完全な回復ならしないほうがましだからです。

●コンパイル・リストの印刷は必要ありません。あれば便利ですが，ソース・プログラムの印刷は他の方法で替えることができるからです。

3-9 コンパイラの作成手順

コンパイラもプログラムなので，作成手順も他のプログラムと同じです。コンパイラに限らずプログラムの作成は，打ち込み，デバッグといったコンピュータに向かう作業より，それ以前の設計段階に念を入れることが大切です。

前に文法書やコンパイラの仕様について述べましたが，コンパイラはこれらをもとに設計します。プログラムの設計では，コンパイラ全体および各処理ルーチンの設計はもちろんですが，その設計の前に，変数や配列へのアドレスの割りつけかた，記号表の形式と検索方法，サブルーチンや関数との引数の受け渡し方法，生成するオブジェクト・コードの種類とランタイム・ルーチンの仕様，エラーの処理方法などをまとめ，プログラム設計書として残しておきます。

特に，オブジェクト・プログラムの変数，配列，引数などのデータ構造，オブジェクト・コードの種類，ランタイム・ルーチンの仕様の決定を初めに行います。これらはコンパイラの性能を決める大切な要素なので，十分に時間をかけてください。

データ構造の設計では，変数や配列のアドレスの割りつけかたやアクセス法，サブルーチンや関数との引数の渡しかたなどの，生成するオブジェクト・プログラム内で使われるデータの構造を決定します。オブジェクト・コードの決定ではまず個々の構文が機械語ではどのようになるのかを考え，次に他の構文と組み合わせることを考えます。このとき，コードが冗長になるようなら最適化の方法も考え，改良するようにします。CPU の命令の関係からどうしても冗長になる場合は，サブルーチン化して，ランタイム・ルーチンとして呼び出すようにしま

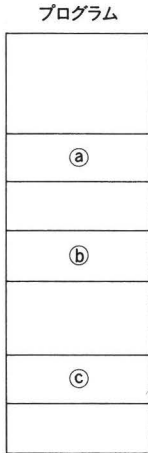
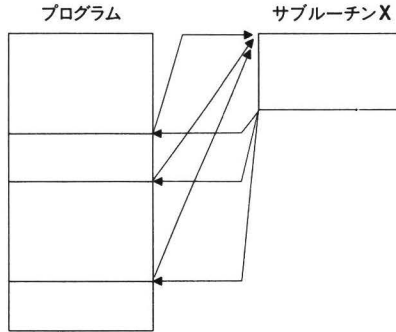
す。図 3－68はサブルーチン化すればどの程度メモリが節約できるかを示しています。ランタイム・ルーチンにまとめるときは，ルーチン名やエントリのラベル，パラメータなどの仕様を書いておきます。

次にコンパイラ内部の各処理の設計を行います。
プログラム設計書が正しくつくられていれば，それにしたがってコーディング，打ち込み，アセンブルあるいはコンパイル，そしてデバッグとなるわけです。デバッグ時には逆アセンブラを備えたモニタあるいはデバッガがあると便利です。それはデバッグ中にオブジェクト・コードをチェックするとき，逆アセンブラがあると生成したコードが一目でわかるからです。デバッグでは多くのテスト・プログラムをコンパイルさせてみることが大切です。それも一つ一つの文から始め，次第に複数の文を組み合わせ，最終的には実用的なプログラムまで，コンパイラが正しくコンパイルするかテストします。できればこの後にサンプルとして他人に評価してもらいましょう。それによって見落としていたミスが発見されることがあります。ここまでくれば一応完成です。

完成したらプログラム設計書，コンパイラの仕様書と文法書をまとめ，コンパイラのプログラム・リストとともに整理し，バグが発見されたときやバージョン・アップのときのため保管しておきましょう。

サブルーチン化とメモリの節約

① サブルーチン化しない場合

同じ内容のルーチンがⓐ, ⓑ, ⓒの3ヵ所で使われている。

② サブルーチン化した場合

ⓐ, ⓑ, ⓒの内容をサブルーチン化し，そのサブルーチンを呼び出すようにプログラムをすると，使用するメモリの量が少なくなる。

節約されるメモリの量は次のような式で求まる。

s ：サブルーチン化するルーチンの大きさ(バイト数)
n ：プログラム中で使用されている回数
c ：サブルーチンの呼び出し命令の大きさ(バイト数)
r ：サブルーチンから戻るための命令の大きさ(バイト数)
m ：節約できたメモリの大きさ(バイト数)

$$m = (s \times n) - (s + r + c \times n)$$

もし，$m \leqq 0$ ならサブルーチン化することでかえって余分なメモリを使うことになる。

第 2 部

第1章

プログラム言語Stellar

第2部では，プログラム言語Stellarのコンパイラを公開します。まず第1章で，Stellarの特徴や設計思想，文法について述べます。

1-1 プログラム言語Stellarとは

Stellarは小規模プログラムの開発用言語で，文法的にはCとPascalを合わせたような構造をしており，簡単なゲームや制御用プログラムなどに適用できます。また，Stellarコンパイラのオブジェクト・プログラムはROM化可能です。

Stellarの最大の特徴は，変数のメモリ割りつけ，および演算がすべてバイト（符号なし8ビット）単位であることです。データがバイト長ですむようなプログラムなら，他のコンパイラに比べ小さなオブジェクト・プログラムを生成することができます。

現在このコンパイラはZ80用のものだけで，CP／Mバージョン，PC-8801バージョン，MSXバージョンの三種類があります。本書ではMSXバージョン以外のものを，特にCP／Mバージョンはアセンブル・リストで公開することにしました。

1-2 Stellarの設計思想

Stellarでは次の二点を設計目標にしました。

①コンパイラ自体が小さくなるようにする。

②小規模なプログラム開発に使えるようにする。

Stellarの言語設計，コンパイラの作成にあたっての条件は次の通りです。

●ハードウェアの条件

①CPUはZ80を使う。

②メモリ（RAM）が少ないパソコンでも使える。

③ディスクがなくてもコンパイルできる。

④他機種への移植が簡単にできる。

●ソフトウェアの条件

①実行速度を上げるため，バイト（8ビット長）を基本データとする。

②Stellarのプログラム中に機械語が書けるような文を用意する。

③変数は静的にメモリに割りつけ，アクセス時の余分なアドレス計算がいらないようにする。

④入出力は入出力文によらず，ライブラリを使う。

また，次のような機能を持つようにしました。

①構造化プログラミングのための制御文をつくる。

②プログラム内で使われる変数は全域と局所の二種類を持つようにする。

③再帰的なプログラムがつくれるようにする。

④オブジェクト・プログラムがROM化できるように，ROMの先頭アドレス（オブジェクト・プログラムのスタート・アドレス）やRAMのアドレス（変数やスタックのエリア）を自由に設定可能にする。

⑤メモリ64Kバイト，I/O 256バイトは何らかの方法でアクセスできるようにする。

1-3 Stellarの構文と文法

ここでは，Stellar の文法を次の五つに分けて解説します。

◆ Stellar プログラムの構成要素
◆プログラムの構造
◆文
◆式
◆変数と定数

Stellar の言語的な特徴は以下の通りです。

①英小文字ベースでプログラムを書く。
②サブルーチンはなく，すべて関数として記述する。
③関数は値を戻さなくてもよい。
④式の中で変数への代入ができる。また，代入をしない式も書ける。
⑤CPU内のレジスタ(B, IX, IY)を使うことができる。

〔1〕 Stellar プログラムの構成要素

Stellar のプログラムは，予約語，名前，数値，定数，文字定数，文字列定数，特殊記号，それに分離符と注釈の八つで構成されています。

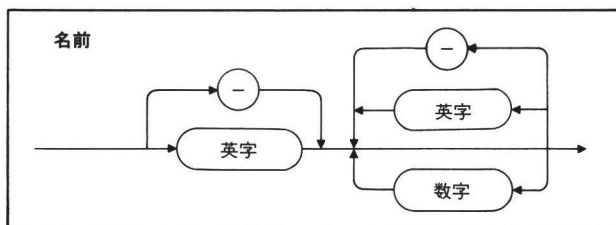
予約語

Stellar は59個の予約語を持っています(**表1-1**)。英小文字でも英大文字でも同じ意味になります。たとえば〝WHILE〟，〝WHiLe〟，〝WhiLe〟はいずれも予約語〝while〟と判断されます。

名前

名前は変数や定数，関数などを表し，構文は次のようになります。

表1-1 stellarの予約語

and	dex	inc	minus	rl	to
at	debug	include	not	rlc	troff
break	else	inline	or	rr	tron
by	elseif	inx	overflow	rrc	until
byte	exit	ix	parity	set	var
carry	for	iy	plus	sign	while
cons	go	ldx	port	sra	word
data	goto	loop	prog	stop	xor
dec	hi	low	recursive	stx	zero
decj	if	memory	return	then	

名前は英数字（A～Z，a～z，0～9）とアンダースコア（_）からなる，数字以外の文字で始まる文字列です。長さに制限はありませんが，意味を持つのは最初の12文字までです。

●正しい名前の例

ABC

__Ad0123

abcdefghijklmn　…初めの12文字(abcdefghijklまで）が名前となる。

ab__XYZ

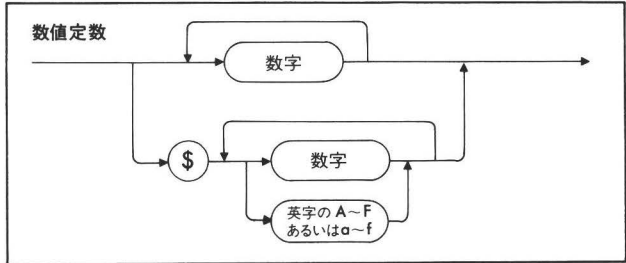
●誤った名前の例

1ABC　…数字で始まっている。

ab@cd　…名前の中に@(英数字でも_でもない）がある。

low　　　…予約語は名前として使えない。

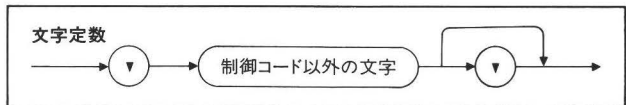
数値定数

数値定数には10進数と16進数があり，次のような構文で表されます。

10進数は数字（0～9）だけで構成された文字列で，0～65535までの値でなければなりません。16進数はドル記号（$）で始まる数字と英字A～Fあるいはa～fで構成された文字列で，16進数で0～FFFFまでの値でなければなりません。

例

10 進 数	0	120	0043	1023	32934	255
16 進 数	$0	$78	$002B	$3FF	$80a6	$ff

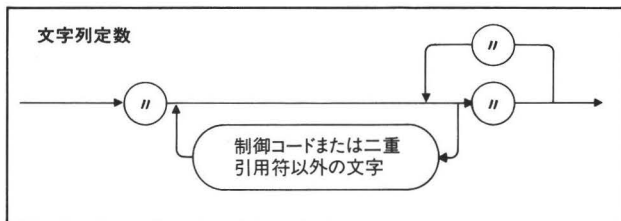
文字定数

文字定数は文字自身をデータとします。

文字定数は制御コード以外の1文字を引用符（'）で囲んだもので，文字コードを数値として扱います。右側の引用符は省略することができます。

例

'A'…$41と同じ　　'm'…$63と同じ
'ア'…$B1と同じ　　'''…$27と同じ
'A…$41と同じ　　'¥…$5Cと同じ

文字列定数

文字列定数は文字列自身をデータとしてメモリ上に格納する場合に使います。

文字列定数は二重引用符（″）で囲まれた0文字以上の文字列のことです。二重引用符を二つ続けて書けば文字列の中に一つの二重引用符を入れることができます。

例

″ABC012″ ……………文字列ABC012がメモリに格納される。

″ABC″″abc″ …………文字列ABC″abcがメモリに格納される。

″″ ………………………空の文字列，何も格納されない。

″カナ　漢字″…………カナ文字，漢字も使用できる。ただし漢字はシフトJISコードだけを使う。

特殊記号

特殊記号には33個の演算子と区切り記号があります（表1－2）。

表1-2 特殊記号

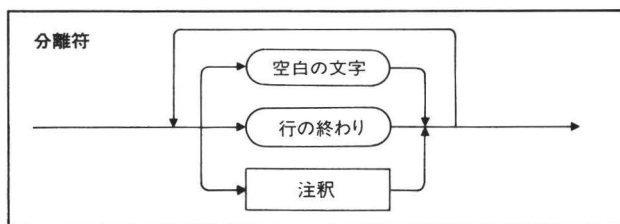
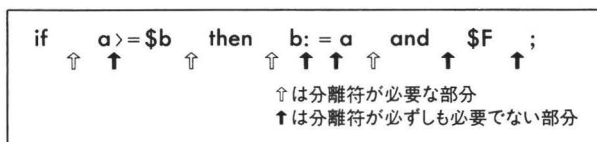
!	(	,	:	<<	>	@	^	~
#	)	−	:=	<=	>=	@@	{	
%	*	.	;	<>	>>	[	¦	
&	+	/	<	=	?	]	}	

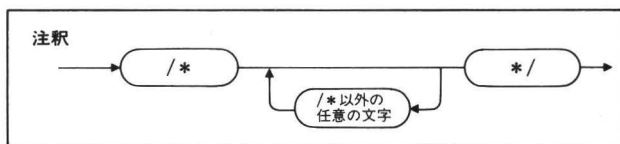
分離符

分離符には空白文字(タブも空白)，行の終わり，注釈の三つがあり，予約語，名前，定数，特殊記号を区切るために用いられます。これらが構文的に明確に識別できる場合は省略できます。

例

```
if     a>=$b     then     b: = a     and     $F     ;
    ⇧ ⬆       ⇧        ⇧ ⬆ ⬆ ⇧          ⬆        ⬆
```

⇧は分離符が必要な部分
⬆は分離符が必ずしも必要でない部分

注釈

注釈は／*で始まり*／で終わる文字列です。*／の2文字をこの順で注釈の文字列中に書くことはできません。

注釈はコンパイル，実行に何ら影響を与えません。

〔2〕プログラムの構造

Stellarのプログラムは主プログラムといくつかの関数からなっています（図1−1）。これを構文図で表すと次のようになります。

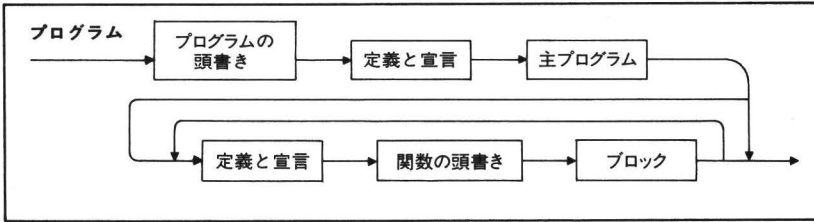
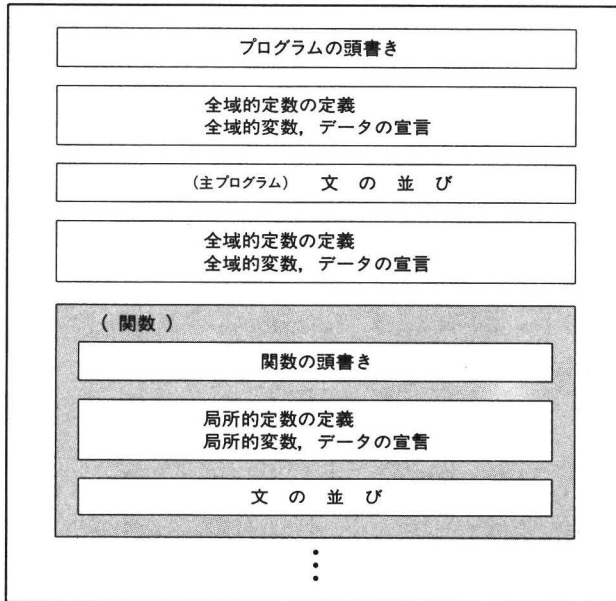
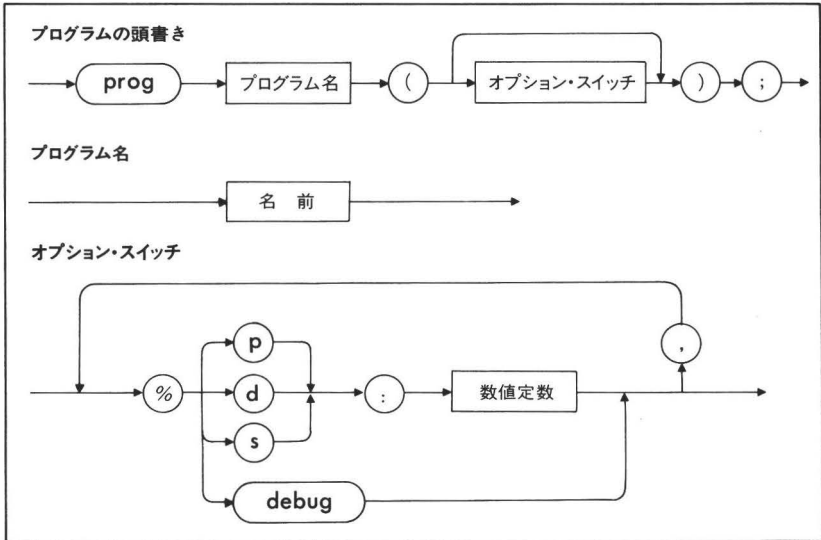



図1-1 stellar プログラムの構造

(1)プログラムの頭書き

Stellarのプログラムは必ず頭書きから始まらなければなりません。構文は次の通りです。

頭書きではプログラムに名前をつけ，コンパイラに対する指示を与えます。この指示はオプション・スイッチといい，次の四つがあります（英字は小文字でも大文字でも可）。

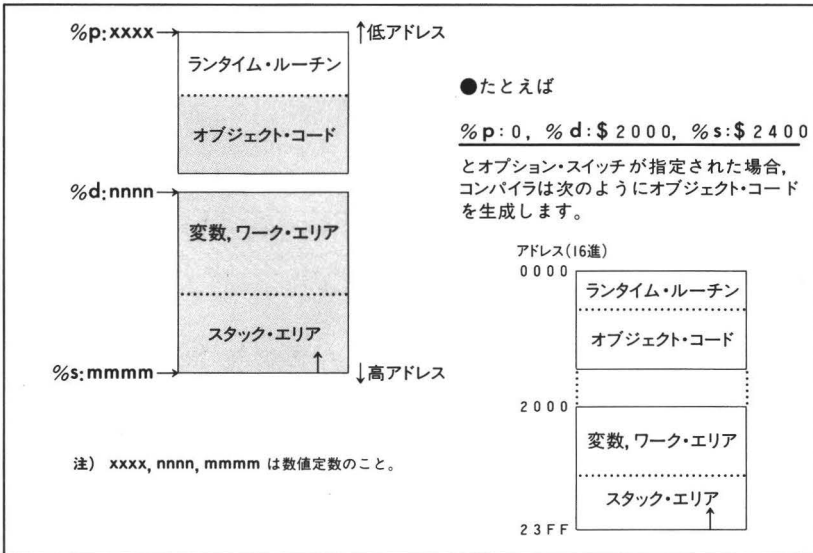
%p：アドレス　…オブジェクト・コードのロード開始アドレスを指定する。

%d：アドレス　…変数やワーク・エリアの先頭アドレスを指定する。

%s：アドレス　…スタックのボトム・アドレスを指定する。

% debug　…デバッグ・モードでコンパイルする。

%p, %d, %sはコンパイル時のアドレスを指定するもので，各アドレスは**図1－2**のようにします。指定されないとデフォルトのアドレスとなります。デフォルト値は機種ごとに異なるので注意してください。たとえばCP／Mバージョンでは%p：＄0100，%d：＄4000で%sが

図1-2 オプション・スイッチ%p, %d, %sとオブジェクト・プログラム

6, 7番地の内容となります。

% debugを指定しないと, コンパイラはデバッグのための文を注釈として扱います。

例

prog TEST＿I ( );

…プログラム名はTEST＿I, オプション・スイッチの指定なし。

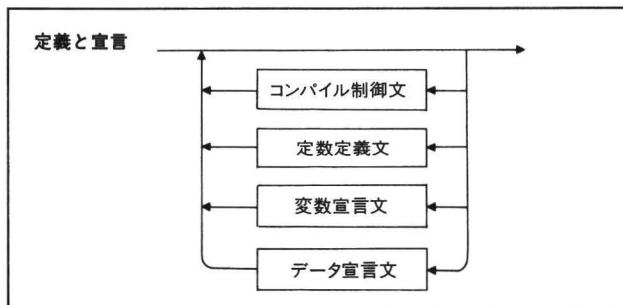
prog utl(%p : $1000, %d : $4000, %s : $9000) ;

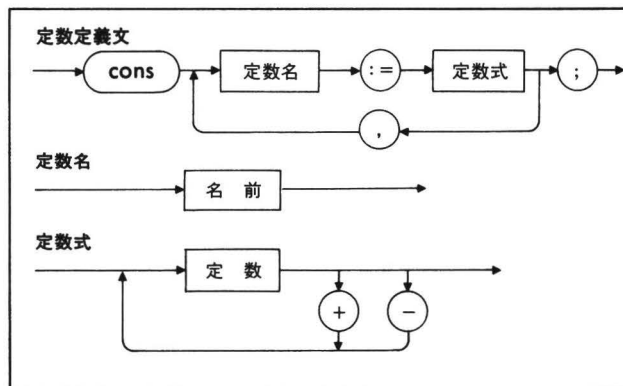
…プログラム名はutl, オブジェクト・コードは16進数で1000番地から, 変数, ワーク・エリアは4000番地, スタック・ボトムは9000番地。

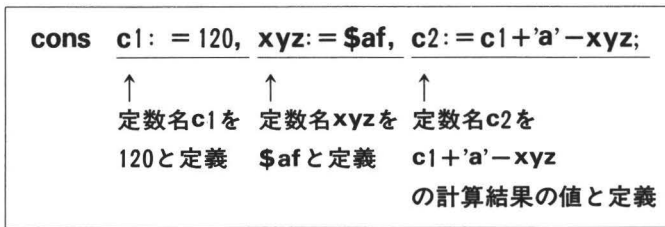
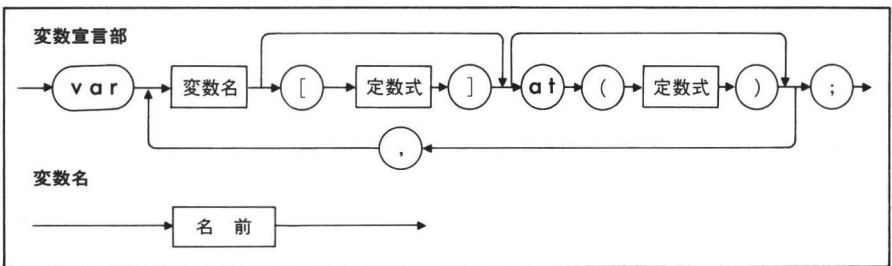
prog fxn(% debug, %d : 12288) ;

…プログラム名はfxnでコンパイルはデバッグ・モード, オブジェクト・コードは12288(16進数では3000)番地から出力。

⑵定義と宣言

定義と宣言は，定数定義部，変数宣言部，データ宣言部，およびコンパイル制御文からなります。

コンパイル制御文については〔3〕で解説していますので，そちらをごらんください。

定数定義部はアセンブラ言語のEQU命令に相当するもので，定義された名前のことを定数名といいます。定数名はプログラム中では数値定数，文字定数と同等に扱われます。

定数名で定義できるのは，定数式で計算された16ビット長の値です。定数式とは定数（数値や文字，あるいは定義ずみの他の定数名など）同士を加減した式のことで，計算は常に16ビット長で行われオーバーフローは無視されます。

例

変数宣言部はプログラム中で使う変数を宣言する部分で，使われる変数はすべてここで宣言しておかなければなりません。

ここでは変数にアドレスを割りつけるだけで，初期値は出力しません。ですから，初期値が必要な変数はプログラム中で値を設定しなければなりません。**図1－3**は変数の宣言とメモリへの割りつけを示した例です。

データ宣言部は定数をメモリ上に記憶するためのもので，記憶した定数にはデータ名がつけられます。データ名は代入できないことを除き，プログラム中で変数と同等に使うことができます。

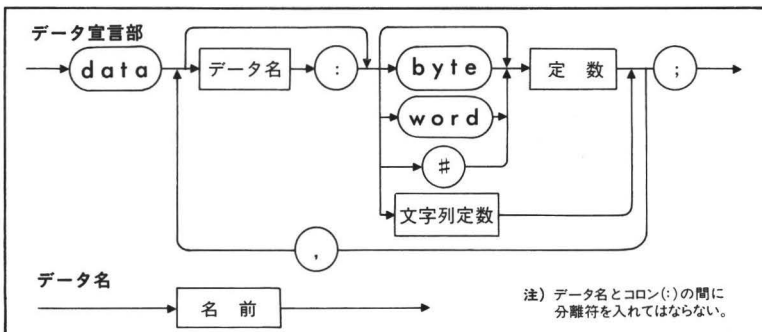



図1-3 変数の宣言とメモリへの割り付け例

変数の宣言

```
Var a, b at($5c), c[0], d[2], e[1], f[3] at($80), g[5], h ;
```

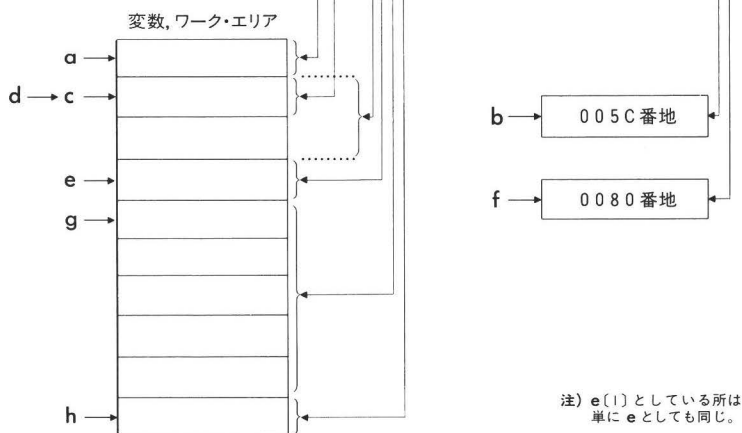
メモリへの割り付け状態

変数，ワーク・エリア

a →
d → c →
e →
g →
h →

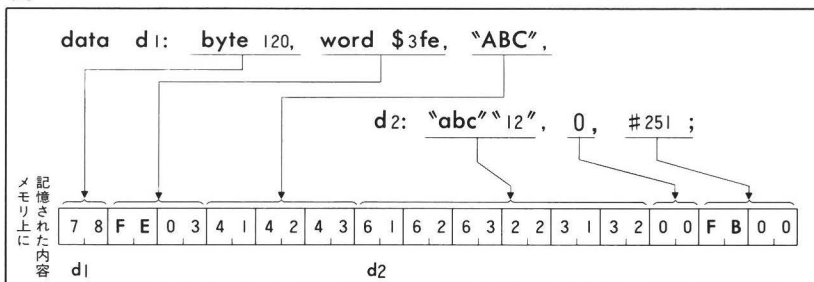
b → 005C番地

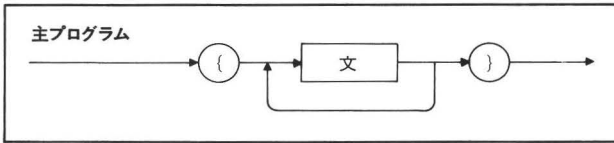
f → 0080番地

注）e〔1〕としている所は
単に e としても同じ。

●at の付かない変数は%d で設定したエリアの方に取られる。一つの変数の大きさはすべて1バイトで，〔定数式〕で定数式の計算結果の長さ（バイト数）だけの領域が取られる。

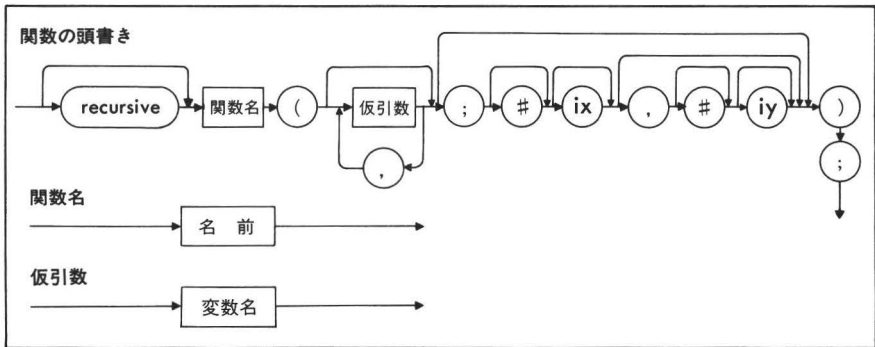
●at の付いた変数は指定されたアドレスを変数のアドレスとし，変数，ワーク・エリアには領域を取らない。
at の付いた変数に〔定数式〕を書いてもそれは無視され注釈として扱われる。

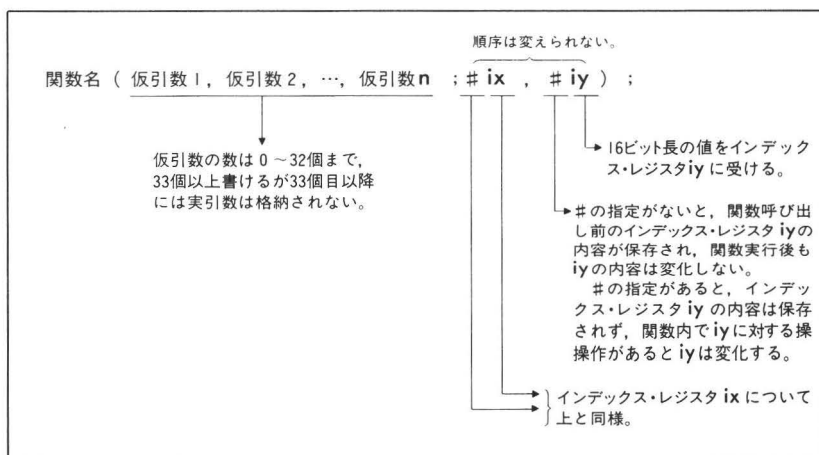
定数の前の byte や word は，記憶がバイト単位なのかワード単位なのかを指定するものです。バイトで記憶する場合は byte の指定は省略できます。また，word の代わりに#を使うこともできます。

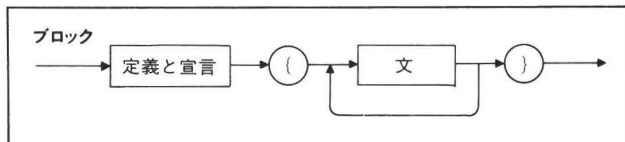
例

⑶主プログラム

Stellarのプログラムは必ず主プログラムから実行されます。

主プログラムは一つの複合文として書かれます。

⑷関数の頭書き

関数の頭書きは，ここから関数が始まることを宣言するためのもので，一つの関数は関数の頭書きとブロックにより構成されています。

再帰的に呼び出す関数は，関数名の前にrecursiveと書かねばなりません。recursiveと書かれている関数は，呼び出されたとき局所的な変数の値をスタックに保存します。

Stellarでは関数に実引数の値を渡します。引数には8ビット長と16ビット長の二種類があり，8ビット長の値は仮引数（局所的な変数）に，16ビット長の値はＩＸあるいはIYのインデックス・レジスタに格納されます。関数の頭書きに書かれる仮引数やインデックス・レジスタ名は次のような意味を持っています。

⑸ブロック

ブロックは，定義と宣言，および一つの複合文により構成されます。

ブロック内で定義あるいは宣言された定数名，変数名，データ名は局所的な名前として扱われ，有効範囲はその関数内だけです。

⑹名前の有効範囲

Stellarでは名前の有効範囲を次のように決めています。

①局所的な名前

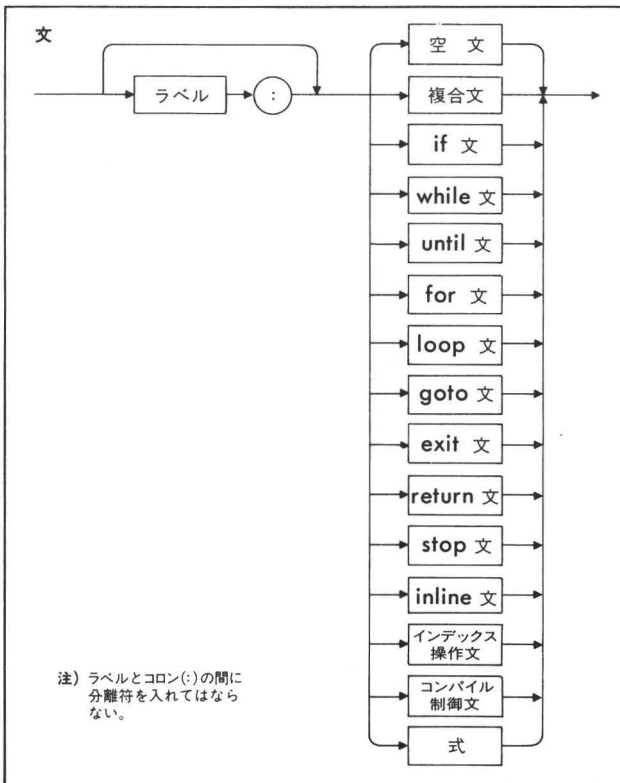
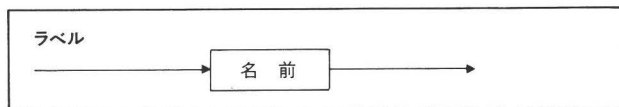
仮引数，およびブロック内で定義された定数名，変数名，データ名は局所的な名前で，同一関数内でのみ有効です。主プログラムや他の関数からは使えません。全域的な名前と同じ名前を局所的な名前として定義，宣言した場合，その関数内では局所的な名前を優先し，全域的

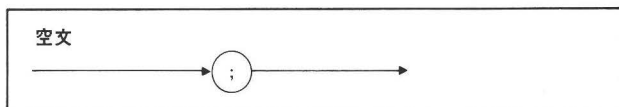
な名前のほうは参照できません。

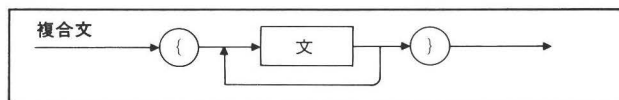
②全域的な名前

局所的な名前以外はすべて全域的な名前です。全域的な名前は，主プログラムや関数の別なくプログラム全体で共通に使えます。全域的な名前には関数名や定数名，変数名，データ名があり，関数名以外は定義，宣言された以後有効になります。関数名については，書かれている場所によらず，すべての場所で有効です。

〔3〕文（ステートメント）

文は実際に計算や仕事をするもので，次の構文に示すように15種類あります。式は次の〔4〕で解説するとして，ここでは式以外の文について解説します。

文にはラベルをつけることができます。ラベルは主プログラムや各々の関数内でのみ有効な局所的な名前です。

(1)空文

何の動作もしない文のことです。

(2)複合文

複合文はいくつかの文をまとめて一つの文として扱うもので，文が書かれている順に実行が行われます。

(3)**if** 文

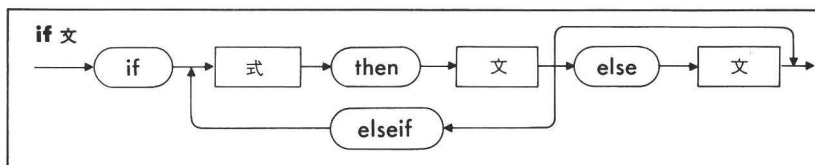
if 文の構文は次の通りです。

if 文は式の計算結果がゼロ以外のときを真，ゼロのときを偽としています。真のときは then の次の文を，偽のときは else の次の文を実行します。else の次の文が if 文のときは，elseif とすることができます。

図 1－4 は if 文のいくつかの形式を流れ図で表したものです。

図1-4 if文

if 式 then 文

式の計算結果は？ ゼロ ゼロ以外 文

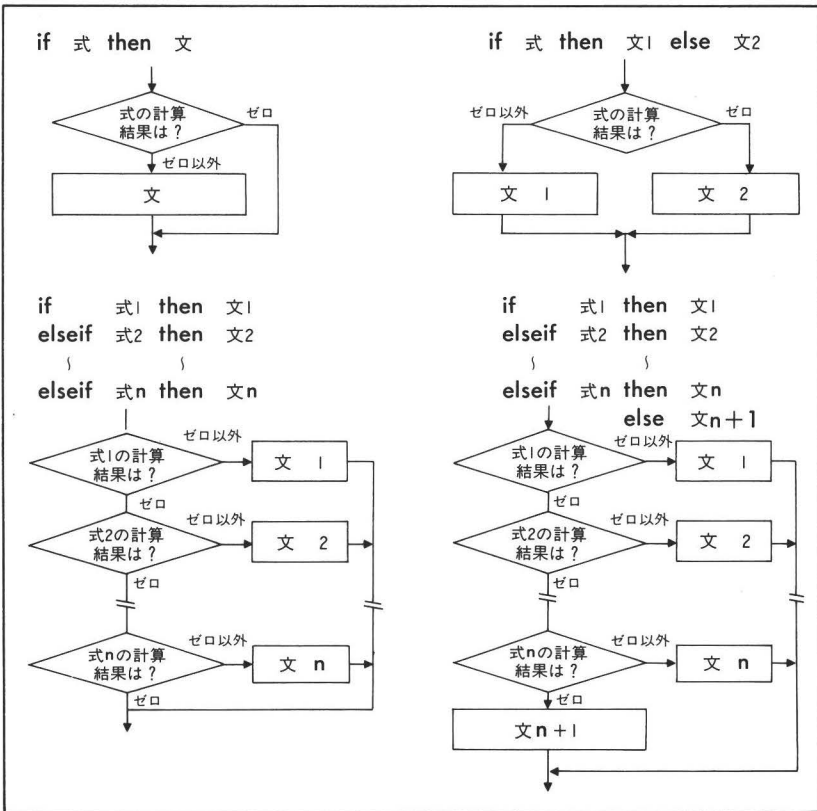
if 式 then 文1 else 文2

式の計算結果は？ ゼロ以外 ゼロ 文 1 文 2

if 式1 then 文1
elseif 式2 then 文2
〜 〜
elseif 式n then 文n

式1の計算結果は？ ゼロ以外 文 1 ゼロ
式2の計算結果は？ ゼロ以外 文 2 ゼロ
式nの計算結果は？ ゼロ以外 文 n ゼロ

if 式1 then 文1
elseif 式2 then 文2
〜 〜
elseif 式n then 文n
else 文n＋1

式1の計算結果は？ ゼロ以外 文 1 ゼロ
式2の計算結果は？ ゼロ以外 文 2 ゼロ
式nの計算結果は？ ゼロ以外 文 n ゼロ
文n＋1

(4) while 文

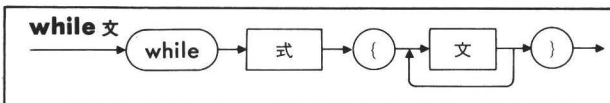
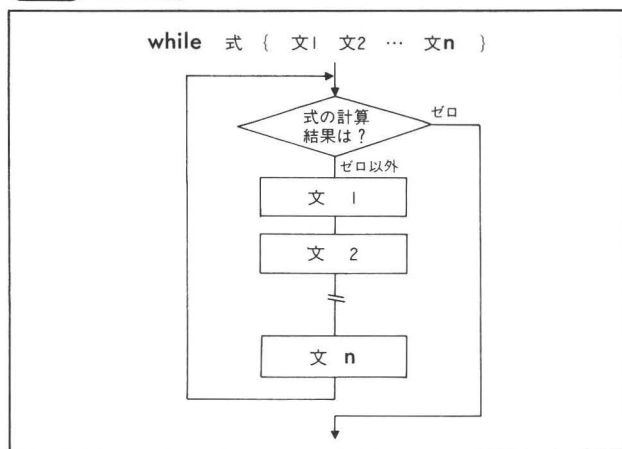
繰り返しのための文の一つで，構文は次のようになっています。

while 文は，式の計算結果がゼロになる（偽になる）まで式の後ろの複合文を繰り返し実行します。while 文では始めに式の計算結果を調べるため,場合によっては後ろの複合文が一度も実行されないことがあります。

図1-5 **while文**

(5) until 文

until 文も繰り返しを行うための文で，構文は次のようになっています。

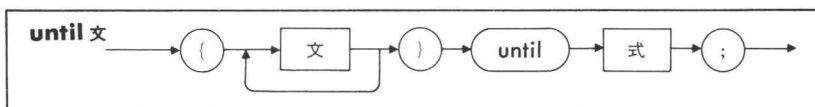
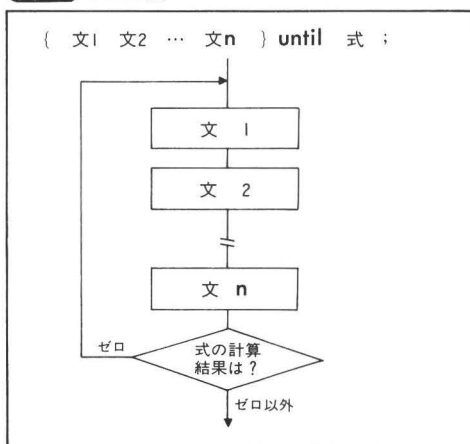



図1-6 **until文**

until 文はまず until の前の複合文を実行し，その後に式の計算を行います。そして結果がゼロ(偽)ならさらにもう一度 until の前の複合文を実行し，結果がゼロ以外（真）なら繰り返しを終えます。

(6) for 文

for 文の構文は次の通りです。

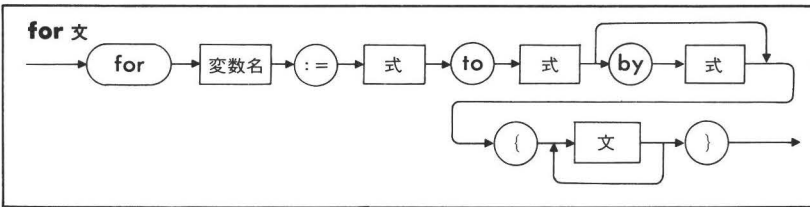
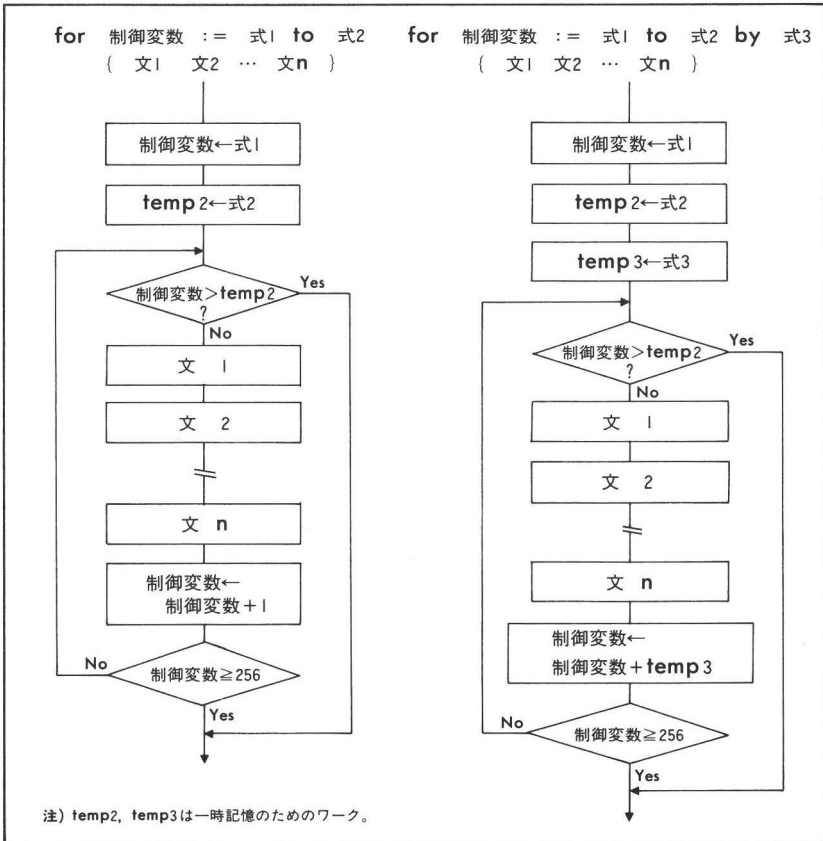
図1-7 for文

for 制御変数 := 式1 to 式2
{ 文1 文2 … 文n }

制御変数←式1
temp2←式2
制御変数>temp2?
Yes
No
文 1
文 2
文 n
制御変数←制御変数+1
制御変数≧256
No
Yes

for 制御変数 := 式1 to 式2 by 式3
{ 文1 文2 … 文n }

制御変数←式1
temp2←式2
temp3←式3
制御変数>temp2?
Yes
No
文 1
文 2
文 n
制御変数←制御変数+temp3
制御変数≧256
No
Yes

注）temp2, temp3は一時記憶のためのワーク。

for 文は流れ図では**図1-7**のようになります。for の次の変数を特に制御変数といいます。：＝の後の式を初期値，to の後の式を終値として，制御変数が

制御変数の値　＞　終値

あるいは

制御変数の値　≧　256

となるまで繰り返します。このとき，ｂｙ式の増分が指定されていれば制御変数にその増分を加算し，増分の指定がなければ制御変数を＋１します。

for 文では初期値，終値，増分を示す各式を繰り返しの前に計算します。初期値は制御変数に代入され，終値，増分はメモリ上の一時記憶領域に記憶します。繰り返しの途中はその領域に記憶した値を使います。

Stellar の場合，変数は１バイト長だけなので，繰り返し回数の最大は

for 変数名：＝ 0 to 255

としたときの256回です。

(7) loop 文

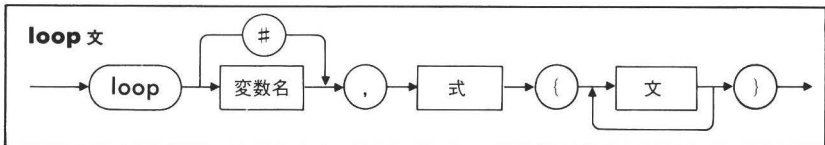
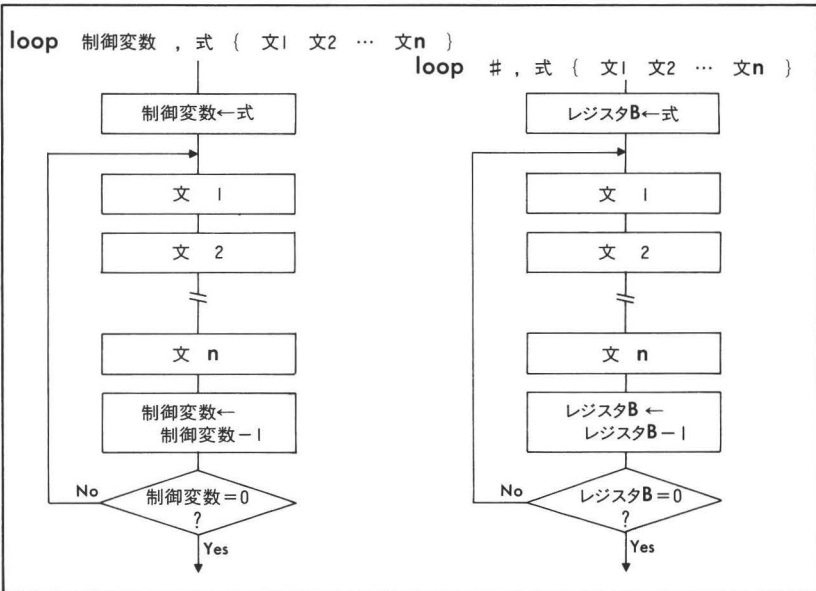
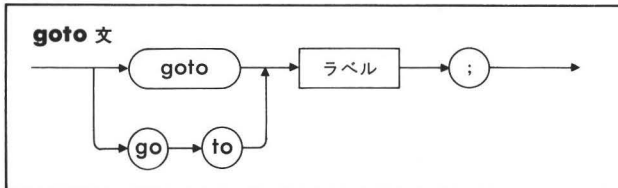
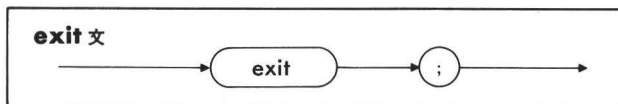
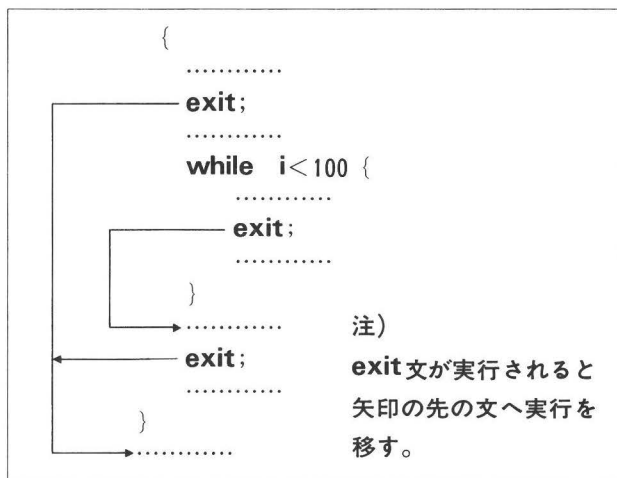
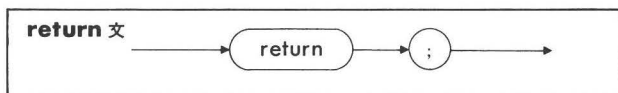
loop 文は指定された回数を単純に繰り返すだけの文で，構文は次のようになっています。

loop 文を流れ図で表すと**図 1－8** のようになります。単純な繰り返しなら for 文よりも loop 文のほうがコンパイル後のオブジェクト・コードが少なくてすみます。特に制御変数名の代わりに#を使うとＢレジスタでループするようなオブジェクト・コードがつくられるため，さらに効率がよくなります。

図1-8 **loop文**

(8) **goto 文**

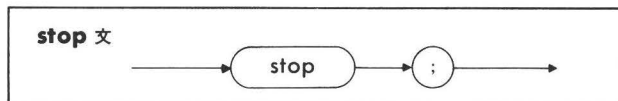
ラベルが書かれている文に実行を移します。

(9) **exit 文**

複合文や while 文，until 文，for 文，loop 文の繰り返しから抜け出るときに使います。

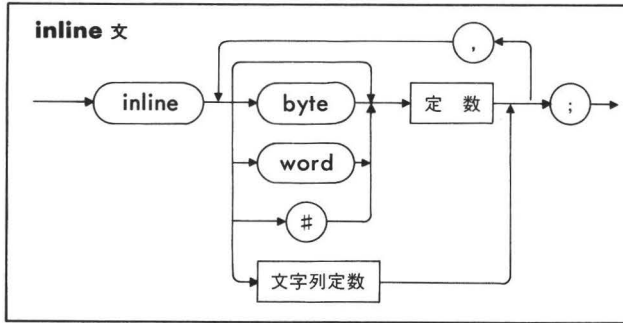
例

⑽ **return 文**

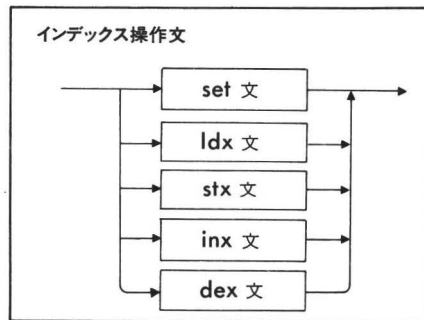
関数の実行を終わり，呼び出した主プログラムや関数に戻るときに使います。

関数の最後の return 文は省略しても自動的に挿入されます。また，主プログラム内の return 文は stop 文と同じ意味です。

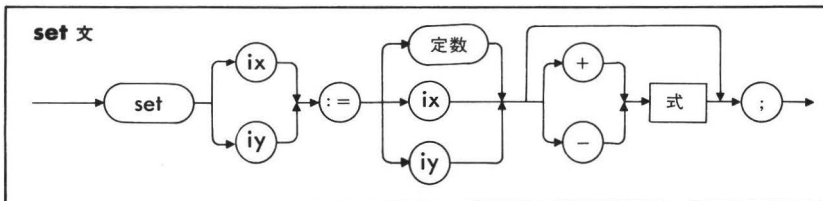
⑾ **stop 文**

プログラムを終了させるための文です。

主プログラムの最後の stop 文は省略しても自動的に挿入されます。

⑿inline 文

inline 文はプログラム中に機械語命令やデータを入れるための文で，構文は次の通りです。

⒀インデックス操作文

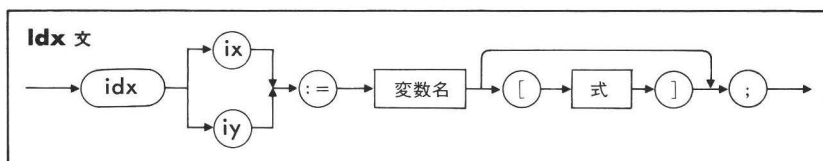
インデックス操作文は二つのインデックス・レジスタ IX, IY を操作するための文で，次に示すような五つの文があります。

① set 文

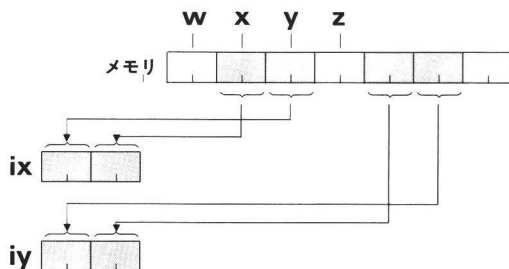
set 文はインデックス・レジスタに定数や値をセットする文です。

定数とインデックス・レジスタは16ビット長で扱われます。式は 8 ビット長で計算され，インデックス・レジスタの値（16ビット長）に加算または減算されます。

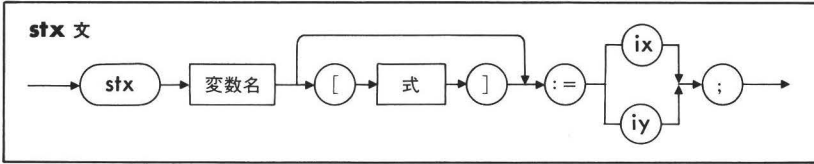
例

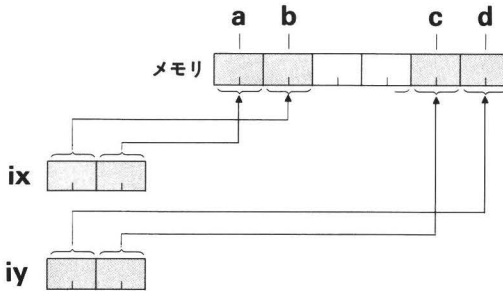
```
set ix : =$3F2C ;
set iy : =258+a/3 ;
set ix : =iy−b * 2 ;
set ix : =ix+3 ;
set iy : =ix ;
```

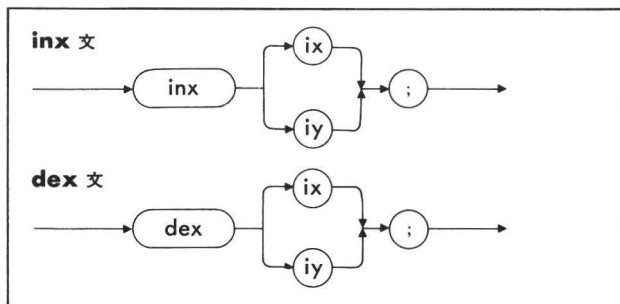
② ldx 文

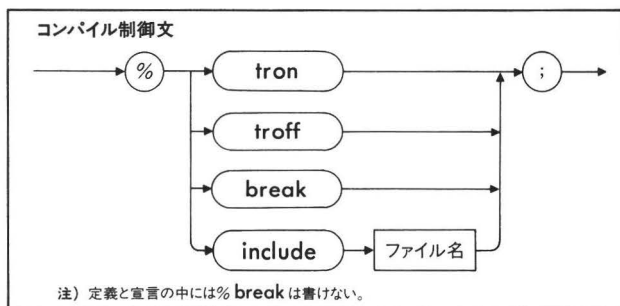
ldx 文はインデックス・レジスタに変数の値をロードする文です。

この文によりインデックス・レジスタの下位8ビットには変数の値が，上位8ビットには変数の次のアドレスの値がロードされます。ldx 文と次の stx 文だけが変数を16ビット長で扱います。

例

③ stx 文

stx 文はインデックス・レジスタの値を変数にストアする文です。

この文によりインデックス・レジスタの下位8ビットが変数へ，上位8ビットが変数の次のアドレスへストアされます。

例

④ inx 文，dex 文

inx 文は指定したインデックス・レジスタの値を＋1し，dex 文は－1します。

⒁コンパイル制御文

コンパイル制御文には，%tron, %troff, %break, %include の四つの文があります。

%tron, %troff, %break はプログラムのデバッグのための文で，デバッグ・モードでコンパイルしたときだけ有効になります。このうち，%tron, %troff はプログラムのトレースに関する文で，%tron を指定するとプログラムの実行状態が追跡できるようなオブジェクト・コードを生成し，%troff が指定されるまで続きます。%tron によって生成されるコードは，実行した行番号あるいは関数名を表示するもので，行番号は

［行番号］

関数名は

{関数名}

と表示します。

%break はプログラムを一時的に中断する文で，これを実行すると

** break in 行番号

と表示して実行を中断します。中断したプログラムは再開させることができます。%tron, %troff, %break は現在のところ，PC-8801バージョンとMSXバージョンだけでサポートしています。

%include は指定されたファイルをソース・プログラムの一部として読み込むための文で，ライブラリを読み込むのに使います。%include は現在，CP／Mバージョンだけでサポートする機能で，ファイル名はCP／Mの規則に従います。PC-8801バージョン，MSXバージョンで% include を使うとエラーになります。

〔4〕式

Stellarの式はすべて8ビット長で演算されます。演算は論理，関係，算術の三つに分類され，次のような構文になっています。

(1)演算子と演算

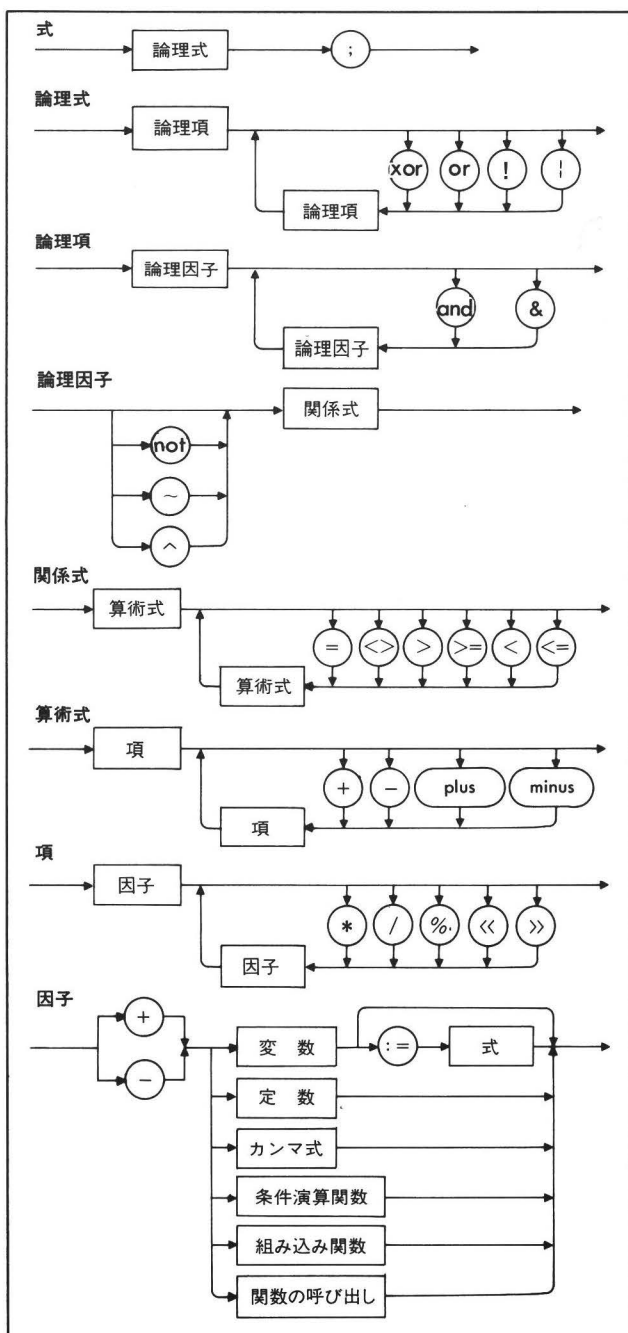
演算子は25個あり，それぞれ次に示すような演算を行います。

①論理演算子

論理演算子はブール演算やビット操作を行います。

or ! ¦	**論理和**
xor	**排他的論理和**
and &	**論理積**
not ~ ^	**論理否定**

論理演算は各ビットごとに演算が行われます。

式
論理式
;
論理式
論理項
xor
or
!
!
論理項
論理項
論理因子
and
&
論理因子
論理因子
関係式
not
~
^
関係式
算術式
=
<>
>
>=
<
<=
算術式
算術式
項
+
−
plus
minus
項
項
因子
*
/
%
<<
>>
因子
因子
+
−
変 数
:=
式
定 数
カンマ式
条件演算関数
組み込み関数
関数の呼び出し

②関係演算子

関係演算子は二つの値を比較するためのものです。

演算子	意味
＝	等しい
＜＞	等しくない
＞	大きい
＞＝	大きいか等しい
＜	小さい
＜＝	小さいか等しい

関係演算子は符号なしの2進整数で比較を行い，結果が真のとき255，偽のとき0の値が得られます。

③算術演算子

算術演算子には次のものがあります。

演算子	意味
＋	加算
－	減算
単項 ＋	2の補数はとらない(何もしない)
単項 －	2の補数をとる
＊	乗算
／	除算
％	剰算
plus	キャリーフラグを含めた加算
minus	キャリーフラグを含めた減算

算術演算は符号なしの2進整数で行われます。

④その他の演算子

演算子としてはその他に，シフトと代入があります。

演算子	意味
＜＜	左シフト（＜＜の右辺の値で左辺の値を論理左シフトする）
＞＞	右シフト（＞＞の右辺の値で左辺の値を論理右シフトする）
：＝	代入（：＝の右辺の値を左辺へ代入する）

(2)演算の優先順位

演算は次の優先順位によって行われます。

高

1．（　）や［　］の中の式，：＝の右辺，関数

2．単項の＋　－

3．＊　／　％　＜＜　＞＞

4．＋　－　plus minus

5．＝　＜＞　＞　＞＝　＜　＜＝

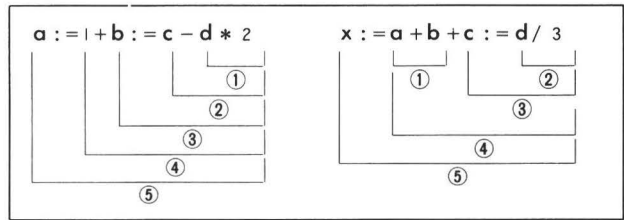
6．not

7．and

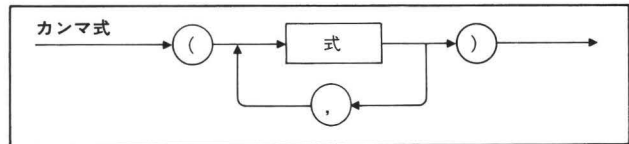
8．or xor

低

同順位の演算は左から右へ行われます。

例

演算は①→⑤の順に行われます。

(3)カンマ式

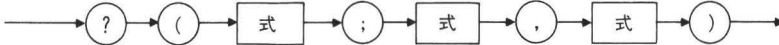
カンマ式の構文は次の通りです。

カンマ（　,）で区切られたカッコ内の式は左から右へ計算され，最右の式の値がカンマ式の値となります。

例

```
x: = (a: =1, b: =5, a+b)
```

aに1, bに5を代入し, a+bを計算する。
a+bがカンマ式の結果となりxに代入される。つまり, xには6が代入される。

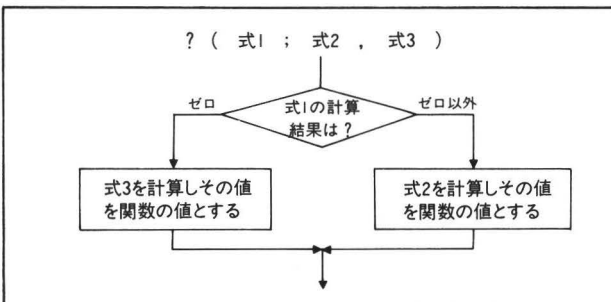
(4)条件演算関数

条件演算関数とは式の中に書けるｉｆ文というようなもので，次のような構文をしています。

カッコ内には三つの式が書かれ，セミコロン（;）とカンマ（ ,）によって区切られます。第一の式が条件を表す式で，計算結果がゼロ以外（真）のとき第二の式を計算し，関数の値とします。第一の式の計算結果がゼロ（偽）なら第三の式を計算し関数値とします。**図1-9** はこれを流れ図で表したものです。

例

```
hex1:= ?((h:=h  &  $f)<10;h,h+7)+'0'
```

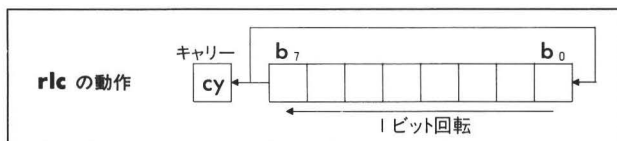
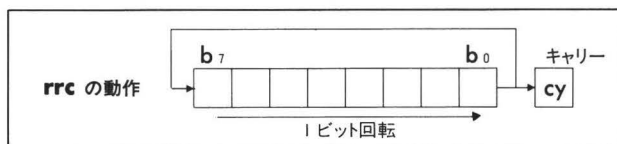
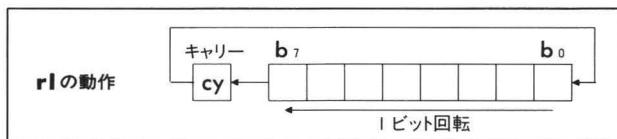
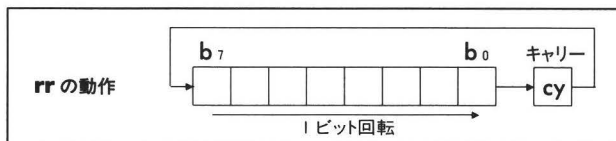
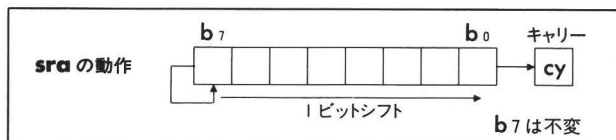
変数hの下位4ビットを1文字の16進数に変換し，hex1へ代入する。

図1-9 条件演算関数

(5)組み込み関数

Stellarでは次の13個の組み込み関数を持っています。

inc(変数名) ………変数の内容を＋１し，＋１した値を関数の値とする。これは（変数名：＝変数名＋１）と同じ。

dec(変数名) ………変数の内容を－１し，－１した値を関数の値とする。これは(変数名：＝変数名－１)と同じ。

rlc(　式　) ………式の計算結果を左へ１ビット回転させる。

rrc(　式　) ………式の計算結果を右へ１ビット回転させる。

rl(　式　)…………式の計算結果を左へキャリーフラグも含めて１ビット回転させる。

rr(　式　) ………式の計算結果を右へキャリーフラグも含めて１ビット回転させる。

sra(　式　)………式の計算結果を右へ1ビット算術右シフトする。

decj(　式　)……式の計算結果を10進数2桁(BCD)に補正する。

carry(　　)………現在のキャリーフラグ(CY)の値を求める。CY＝1なら255，CY＝0なら0が関数値。

carry(　式　)……式計算後のキャリーフラグの値を求める。CY＝1なら255，CY＝0なら0が関数値。

zero(　　)…………現在のゼロ・フラグ(Z)の値を求める。Z＝1なら255，Z＝0なら0が関数値。

zero(　式　)……式計算後のゼロ・フラグ(Z)の値を求める。Z＝1なら255，Z＝0なら0が関数値。

sign(　　)…………現在のサイン・フラグ(S)の値を求める。S＝1なら255，S＝0なら0が関数値。

sign(　式　)……式計算後のサイン・フラグの値を求める。S＝1なら255，S＝0なら0が関数値。

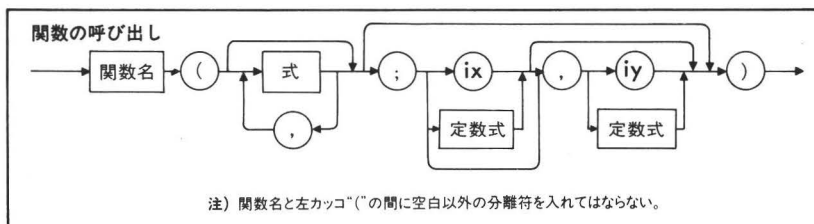
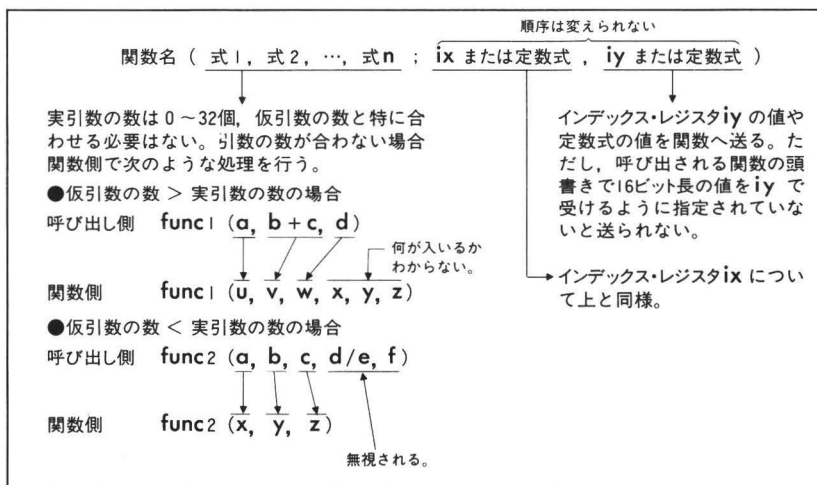
parity(　　)………現在のP／Vフラグの値を求める。P／V＝1なら255，P／V＝0なら0が関数値。

parity（ 式 ）…式計算後のP／Vフラグの値を求める。P／V＝1なら255，P／V＝0なら0が関数値。

overflow（ ）…… parity( )と同じ。

overflow（ 式 ）… parity（ 式 ）と同じ。

(6)関数の呼び出し

関数の呼び出しの構文は次の通りです。

実引数には8ビット長と16ビット長の二種類があり，実引数の値が関数に渡されます。セミコロン（；）より前が8ビット長の実引数で，0～32個の式が書けます。セミコロンより後が16ビット長の実引数で，インデックス・レジスタ名や定数式を書きます。関数の呼び出しで書かれる実引数は次のような意味を持っています。

(7)関数の値の渡しかた

関数の値は return 文の一つ前に実行した式の値となります。

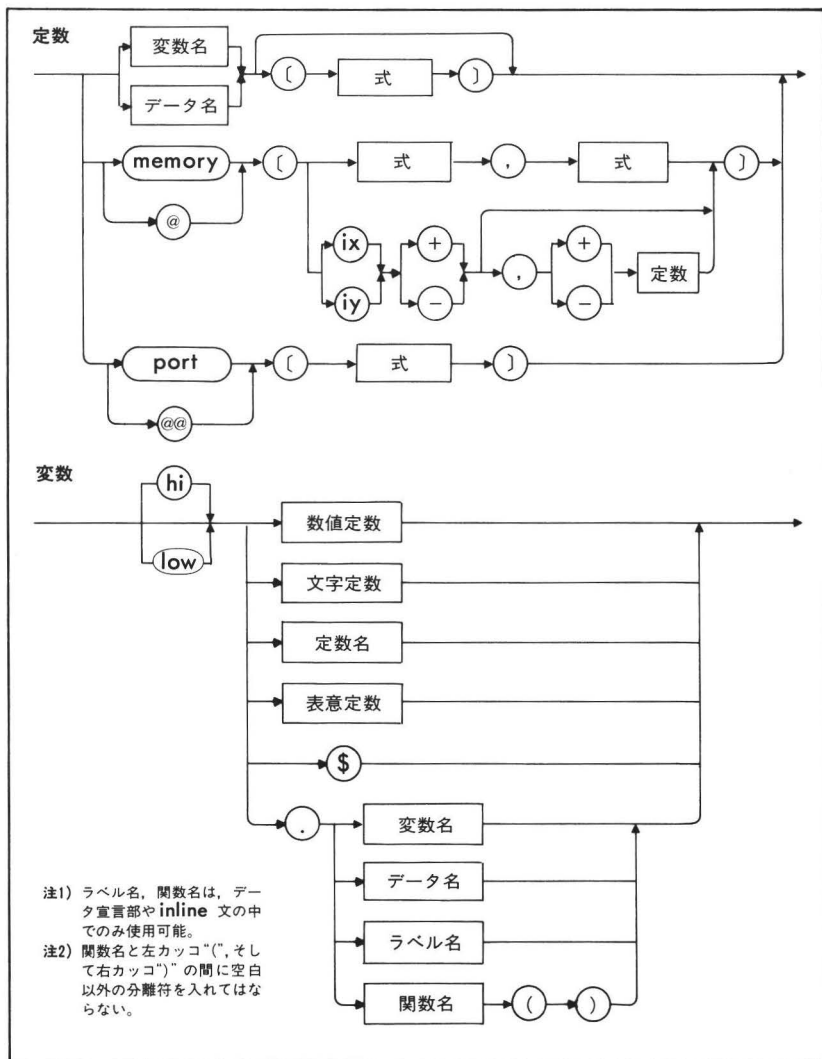
たとえば仮引数 a, b, c の合計を関数の値とする関数は,

```
add3 (a, b, c)
{
    a + b + c;
}
```

となります。次の例は, IX が示す n バイトのメモリの中から x と等しいものを探し出す関数で, 等しいものがあればその位置を, 等しいものがない場合は $FF を関数の値として戻します。

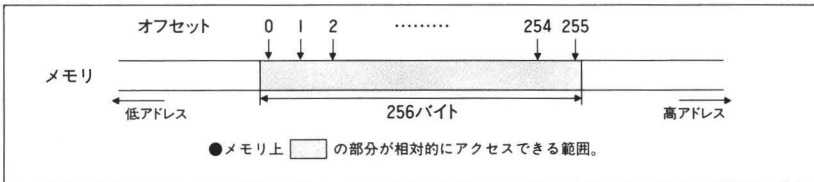
```
sear (n, x ;ix) ;
var i ;
{
    if n=0 then {
      $ ff ;
      return ;
    }
    for i := 0 to n-1 {
      if memory[ ix+] =x then {
        i ;
        return ;
      }
    }
      $ ff ;
 }
```

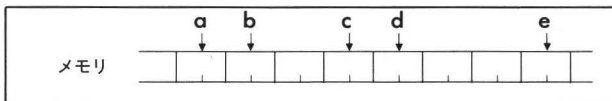
〔5〕変数と定数

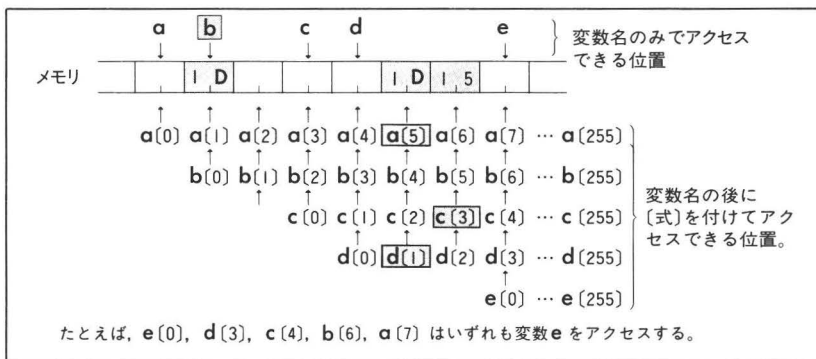
式の中に使われる変数や定数の構文は次のようになっています。

(1)変数

変数として扱うものには，変数宣言部で宣言した変数名，データ宣言部で宣言したデータ名，そしてmemory配列とport配列です。データ名は：＝の左辺に置いて値を代入することはできません。

変数名，データ名の後には［　式　］をつけて相対的な参照，代入（代入は変数名のみ）が行えます。［ ］内の式が変数あるいはデータからのオフセットで0～255までの値となります。図1-10が相対的にアクセスできる範囲を表した図です。

図1-10 相対的にアクセスできる範囲

```
var a,b[2],c,d[3],e;
{
    b:=d[1]:=$1d;
    c[3]:=a[5]-8;
}
```

この例の場合，変数宣言部では変数はメモリ上に次のように割りつけられます。

a, c, eはそれぞれ1バイト，bはb［2］と宣言されているので2バイト，dはd［3］と宣言されているので3バイト取られます。

この例を実行するとメモリ上の値は次のようになります。

memory 配列はCPUの全メモリ空間64Kバイトに対して自由に参照，代入するためのもので，二つの形式があります。

memory [式1，式2]

式1の値を上位バイトとし，式2の値を下位バイトにしてメモリのアドレスを設定し参照，代入を行います。

memory [インデックス・レジスタ名，定数]

この形式はインデックス・レジスタ（IX，IY）を使うものです。基本的な使用法は上記のように，初めにインデックス・レジスタ名（IX，IY）を書き，次に定数を書きます。

インデックス・レジスタ名の後に正符号（+）または負符号（－）をつけると参照，代入を行った後，インデックス・レジスタの値を正符号(+)で＋1，負符号（－）で－1します。

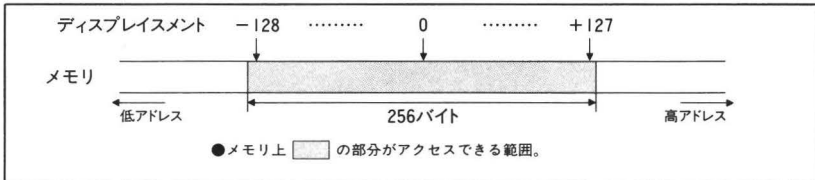
定数は符号をつけない場合で0～127までの整数，符号をつけた場合で－128～＋127までの整数となります。定数の値がゼロの場合，

memory [インデックス・レジスタ名]

と書いてもかまいません。この定数のことをディスプレ

イスメントといい，この定数によりインデックス・レジスタが示すアドレスを中心に－128～＋127バイトがアクセスできるようになります(図1－11)。また，〝memory〟の代わりに@と書くことができます。

port配列は，I／Oポート256バイトに対して参照，代入を行うものです。同様に〝port〟の代わりに@@と書くことができます。

図1-11 ディスプレイスメントとアクセスできる範囲

(2)定数

定数としては次のようなものがあります。

●10進数　0～65535までの正の整数。

●16進数　$を頭につけた0～FFFFまでの16進数。

●文字　1文字を引用符（'）で囲んだもの。

●定数名　定数定義部で定義したとき，値につけた名前。

●アドレス　定数名，データ名，ラベル名，関数名の頭にピリオド（.）をつけるとそのアドレスが定数となります。

●表意定数　コンパイラが自動的に定義する大域的な定数名のことです。表意定数には次の三つのものがあります。

_work	**ランタイム・ルーチンのワーク・エリアの先頭アドレス。**
_var	**変数エリアの先頭アドレス。**
_code	**オブジェクト・プログラムの先頭アドレス。**

●ロケーション・カウンタの値　$の後が16進数以外の文字なら，その$は次の生成するオブジェクト・コードのアドレスを示します。

定数は基本的には16ビットの値を取ります。ただし，式などで使う場合，定数は8ビット長の値（10進数で0〜255，16進数で0〜FF)でなければなりません。もし9ビット以上の値（10進数で256〜65535，16進数では100〜FFFF）だとエラーになります。そのような場合，定数の前にhiまたはlowと書けば上位または下位の8ビットだけを定数とすることができます。

たとえば変数abcのアドレスが16進数でA25Cのとき，low. abcとするとこの値は16進数で5Cとなり，hi. abcでは16進数でA2となります。

1-4 エラーメッセージ

コンパイラ中に表示されるエラーメッセージは，次のとおりです。

〔1〕各バージョン共通のメッセージ

1．行番号：**Missing variable name**：変数名
　内容：指定された変数名が見つからない。
　動作：アドレスがゼロの全域的な変数として処理を続ける。

2．行番号：**Missing constant name**：定数名
　内容：指定された定数名が見つからない。
　動作：値がゼロの全域的な定数名として処理を続ける。

3．行番号：**Bad option switch**
　内容：オプション・スイッチの指定が悪い。
　動作：アボート

4．行番号：**Illegal function name**：名前
　内容：不法な関数名で関数を宣言しようとした。
　動作：名前が予約語や定数ならアボート。二重宣言ならその名前でシンボル・テーブル（記号表）へ再登録し処理を続ける。

5．行番号：**Illegal name**：名前
　内容：不法な名前で定数名，関数名を定義あるいは宣言しようとした。
　動作：名前が予約語や定数ならアボート。二重定義，宣言ならその名前でシンボル・テーブルへ再登録し処理を続ける。

6．行番号：**Illegal label**：ラベル
　内容：不法な名前でラベルを定義しようとした（二重定義など）。
　動作：その名前でシンボル・テーブルへ登録される。

7．行番号：**Bad string data**

内容：文字列定数の指定がない（文字列の右の″がないまま行が終わっている)。

動作：文字列定数の右に二重引用符（″）があるものとして処理を続ける。

8．行番号：**Too many arguments**

内容：実引数の数が32個以上定義されている。

動作：処理を続ける。

9．行番号：**Illegal character**：文字

内容：字句として認められない文字がある。

動作：アボート

10．？：**Undefined label**：ラベル

内容：未定義なラベルがある。

動作：処理を続ける。

11．？：**Undefined function name**：関数名

内容：未定義な関数名がある。

動作：処理を続ける。

12．行番号：**Illegal constant**：定数

　　　″　　　　**displacement**

　　　″　　　　**over range**

内容：不法な定数が指定された。定数値，displacement, over range のいずれかが表示される。

動作：値をゼロにして処理を続ける。

13．行番号：**Bad constant**

内容：定数の指定が悪い。

動作：アボート

14．行番号：**Bad address constant**

内容：アドレス定数の指定が悪い。または指定ができないところでアドレス定数を指定した。

動作：アボート

15．行番号：**Syntax error**

内容：構文が正しくない。その他のエラー。

動作：アボート

16. 行番号：**Bad index operation**
　内容：インデックス操作文が正しくない。
　動作：アボート
17. 行番号：**Bad expression**
　内容：式が正しくない。
　動作：アボート
18. **% Aboat**
　内容：アボート
　動作：コンパイルを異常終了させる。

〔2〕CP／Mバージョンでだけ表示されるエラーメッセージ

　これらのうち，1～7のエラーが発生するとコンパイルを中止する。8，9のエラーは発生するとアボートする。10はエラーではなく警告である。

1．**% No source file**　…該当するソース・ファイルがない。
2．**% No directory space**　…ディスクのディレクトリ領域に空きがないので新しいファイルがつくれない。
3．**% Disk full**　…ディスクに空きがないのでファイルが出力できない。
4．**% Cannot close**　…オープンしたファイルがクローズできない。
5．**% Bad source file**　…入力したソース・プログラムが正しくない。
6．**% Bad file name**　…指定されたファイル名が正しくない。
7．**% Symbol table overflow**　…シンボル・テーブルがオーバーフローした。
8．行番号：**Bad include file name**　…% include で指定されたファイル名が正しくない。
9．行番号：**No include file**　…% include で指定され

たファイルがない。

10. 行番号：**Non supporting(warning)**　**% break**

　　　　　　〃　　　　　　　　**% tron**

　　　　　　〃　　　　　　　　**% troff**

…サポートしていない文を使用した。

〔3〕PC-8801バージョンでだけ表示されるエラーメッセージ

1，2，4のエラーが発生するとコンパイルを中止する。3のエラーが発生するとアボートする。

1．**% Object area full**　…オブジェクト・プログラムを出力する領域がいっぱいになった。

2．**% Symbol table overflow**　…シンボル・テーブルがオーバーフローした。

3．行番号：**Non supporting : % include**　…% include はサポートしていない。

4．行番号：**% Bad source**　…ソース・プログラムの形式が正しくない。

第2章

CP/M バージョンの Stellar コンパイラ

CP/M上で動作するStellarコンパイラの使用法を説明し，全プログラム・リストを公開します。CP/MはV2.2の使用を前提にしています。

2-1 コンパイラの使用法

CP／Mバージョンで Stellar のプログラムを開発するには，次の二つのコマンド・ファイルが必要です。

STELLAR.COM… Stellar コンパイラの本体。

CONVOBJ.COM… コンパイラが出力したオブジェクト・ファイルをインテル HEX形式ファイルに変換する。

図2-1 は Stellar でプログラムを開発する手順を示したものです。ソース・プログラムは CP／M上のエディタで作成します。Stellar コンパイラ(STELLAR.COM)はソース・ファイルを入力してオブジェクト・プログラムを出力します。コンパイラは直接機械語のオブジェクト・コードを出力しますが，特殊なコードを含んでいるため，このままで実行することはできません。そこで，これを実行可能にするためのプログラムが CONVOBJ.COM でインテルHEX形式ファイルあるいはコマンド・ファイルに変換します。

〔1〕STELLAR.COM の実行

Stellar コンパイラはキーボードから次の形式のコマンドを入力して実行します。

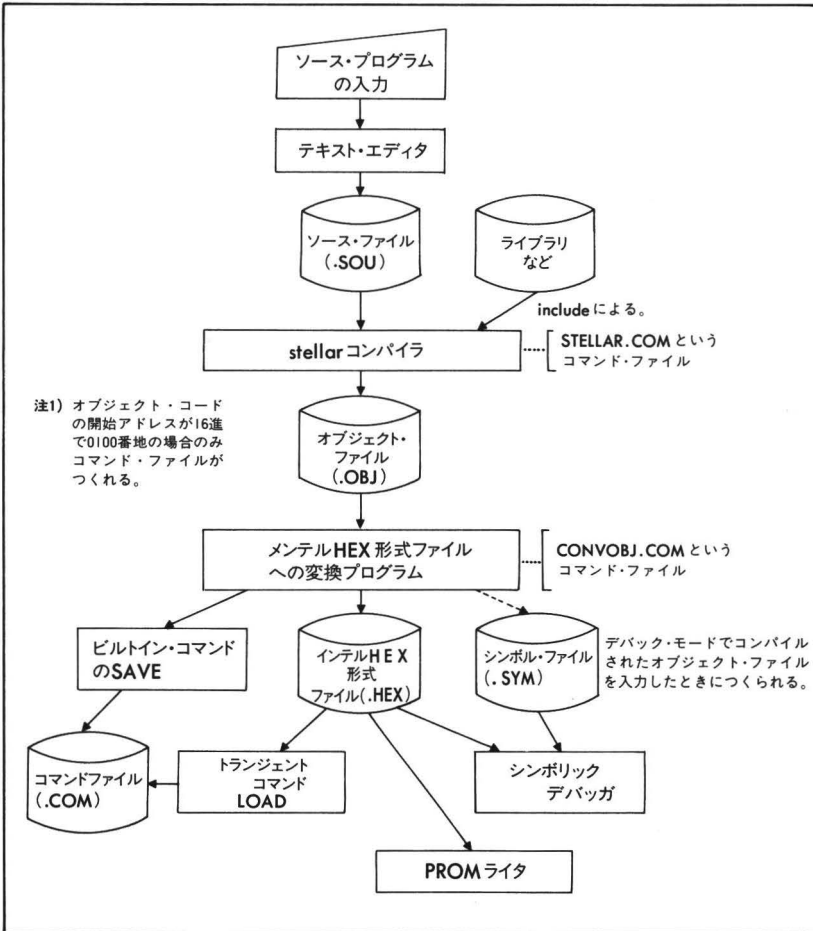
```
STELLAR  [d :] filename [.typ]  [d :] [filename] [.typ] ⏎
           ①      ②        ③       ④       ⑤         ⑥
```

①ソース・ファイルのドライブ名。省略時はカレント・ドライブ名となる。

②ソース・ファイルのファイル名。省略できない。

③ソース・ファイルのファイル・タイプ。省略時は，SOU となる。

④オブジェクト・ファイルのドライブ名。⑤とともに省

図2-1 CP/Mバージョンでのプログラム開発手順

略されている場合はソース・ファイルのドライブ名，④だけの省略の場合はカレント・ドライブ名となる。

⑤オブジェクト・ファイルのファイル名。省略時はソース・ファイルのファイル名と同じ。

⑥オブジェクト・ファイルのファイル・タイプ。省略時は．OBJとなる。

このうち〔　〕で囲まれている内容は省略でき，⏎はリターン・キーを押すことを示しています。また，英大

文字の語はそのまま入力することを，英小文字の語はその語が持つ意味を入力することを示します。ここでは説明上，大文字と小文字を分けていますが，実際の入力では大文字でも小文字でも構いません。

例

B >stellar　XYz ⏎　…ソース・ファイルはカレント・ドライブにある XYZ.SOU という名前のファイル。オブジェクト・ファイルはカレント・ドライブへ XYZ.OBJ という名前で出力。

A >STELLAR　B：ABC.STE ⏎　…ソース・ファイルは Bドライブの ABC.STE という名前のファイル。オブジェクト・ファイルは B ドライブへ ABC.OBJ という名前で出力。

B >stellar sample1 sp1 ⏎　…ソース・ファイルはカレント・ドライブにある SAMOLE1.SOU という名前のファイル。オブジェクト・ファイルはカレント・ドライブへ SP1.OBJ という名前で出力。

C >STELLAR SAMPLE2　B：⏎　…ソース・ファイルはカレント・ドライブにある SAMPLE2.SOU という名前のファイル。オブジェクト・ファイルは B ドライブへ SAMPLE2.OBJ という名前で出力。

注）下線はキーボードからの入力を示します。

Stellar コンパイラは，エラーがなければ次のようなメッセージを表示しながらコンパイルします。この例は **2-2** の **FDUMP.SOU** というプログラムをコンパイルした場合のものです。

```
B>stellar fdump↵

Stellar compiler Rev 1.01 ( CP/M-80,MSX-DOS Version )
Copyright (c) 1984 H.Ohnuki / MIA

 Program  name : file_dump…………………プログラム名
 Function name : puthex    ┐
 Function name : puthexl   │
 Function name : getfile   ├ 関数名
 Function name : bdos      ┘

 Program  036F (0100-046E)…………………生成したオブジェクト・コードのサイズとアドレス
 Data     004B (4000-404A)…………………変数,ワーク・エリアのサイズとアドレス

 ** End of compile, No error(s)
```

〔2〕**CONVOBJ.COM の実行**

CONVOBJ はキーボードから次の形式のコマンドを入力して実行します。

①コンパイラが出力したオブジェクト・ファイルのドライブ名。省略時はカレント・ドライブとなる。

②コンパイラが出力したオブジェクト・ファイルのファイル名。省略できない。

③コンパイラが出力したオブジェクト・ファイルのファイル・タイプ。省略時は.OBJ となる。

CONVOBJ は指定されたファイルに収められたオブジェクト・プログラムを読み込んで，インテル HEX 形式のファイルを出力します。このとき出力ファイルはオブジェクト・ファイルと同一のドライブに，同一のファイル名（ただしファイル・タイプを.HEX）で出力されます。加えて，オブジェクト・プログラムがデバッグ・モードでコンパイルされていたときは，CONVOBJ は同一のドライブ，同一のファイル名（ファイルタイプを.SYM として）でシンボル・ファイルを出力します。シンボル・ファイルには全域的な名前とそのアドレスが出力され，米デジタルリサーチ社のシンボリック・デバッガ（ZSID）で

使える形式になっています。

CONVOBJ はインテル HEX 形式で出力する以外にコマンド・ファイル（ファイル・タイプが.COM）を出力することもできます。ただし，この場合オブジェクト・コードの開始アドレスが16進数で0100番地からでなければなりません。CONVOBJ は終了時にページ数を10進数で表示するので，SAVE コマンドでディスクにセーブするときはその数にしたがいます。

例

B >CONVOBJ　XYZ⏎　…カレント・ドライブにある XYZ. OBJ という名前のオブジェクト・ファイルを入力し，カレント・ドライブに XYZ. HEX というインテル HEX 形式ファイルをつくる。

A >CONVOBJ B : ABC. 080⏎　…ドライブ B にある ABC. 080 という名前のオブジェクト・ファイルを入力し，ドライブ B に ABC. HEX というインテル HEX 形式のファイルをつくる。

CONVOBJ はエラーがなければ次のようなメッセージを表示しながら変換します。この例は 2-2 の FDUMP. OBJ を変換した場合のものです。

```
B>convobj fdump-

  Stellar utility,  convert object ==> intel HEX
  Rev 1.00  Copyright (c) 1984  H.Ohnuki / MIA

  Program  name : file_dump = 0100・・・・・・・・・・・・・・・・プログラムの開始アドレス
  Function name : puthex = 037D
  Function name : puthexl = 03AE
  Function name : getfile = 03EA          } 関数の開始アドレス
  Function name : bdos = 0436
  Constant name : _work = 4000
  Constant name : _var = 4030             } 表意定数の値
  Constant name : _code = 0100
  Constant name : dfcb = 005C
  Constant name : dbuff = 0080
  Constant name : reclen = 0080
  Constant name : putchr = 0002
  Constant name : printf = 0009
  Constant name : constf = 000B           } 全域的な定数名の値
  Constant name : openf = 000F
  Constant name : readf = 0014
  Constant name : on = 00FF
  Constant name : off = 0000
  Variable name : fcb = 005C
  Variable name : buffer = 0080
  Variable name : eof = 4030
  Variable name : bfptr = 4031
  Variable name : d = 4032                } 全域的な変数のアドレス
  Variable name : i = 4033
  Variable name : adr = 4034
  Variable name : ch = 4036
  Data     name : crlf = 020D
  Data     name : nofil = 0210            } 全域的なデータ名のアドレス

  End address    : 046E          ・・・・・・・・・・・・・・・・・・生成したオブジェクト・コードのエンド・アドレス
  Program size   : 036F  [ 4 Page ]・・・・・・・・・・・・・・・・・生成したオブジェクト・コードのサイズ,[ ]内の数がSAVE
                                           コマンドでセーブする場合のページ数(10進数)
```

2-2 サンプル・プログラム

CP／Mバージョンの Stellar のサンプル・プログラムとして，ファイル・ダンプとハノイの塔の二つのプログラムを紹介します。

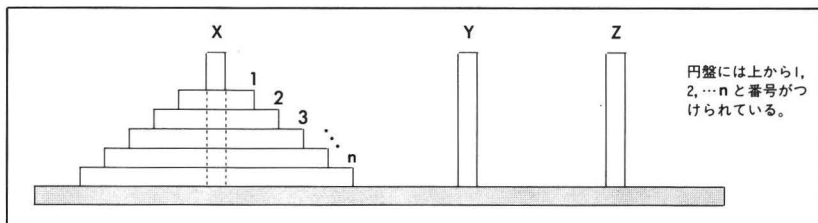
〔1〕ファイル・ダンプ（リスト2－1，実行例2－1）

ファイルの内容を16進数とＪＩＳコードで表示するプログラム。関数 BDOS の使用法に注意してください。

〔2〕ハノイの塔(リスト2－2および2－3，実行例2－2)

再帰的なプログラムと％include の使用例です。**リスト2－3**は HIO. LIB というファイル名にしてください。ハノイの塔は一種のパズルで，下の絵のように三本の塔があり，最初はＸの塔に大きさの順にｎ枚の円盤があります。次の規則ですべての円盤をＺの塔に移します。

（規則１）一度に一枚の円盤しか移動できない。

（規則２）小さい円盤の上に大きな円盤は置けない。

●リスト2-1　ファイル・ダンプ

```
 1:
 2:         /*   file dump   */
 3:
 4: prog    file_dump():
 5:
 6: cons    dfcb    := $005c,       /* default fcb          */
 7:         dbuff   := $0080,       /* default buffer       */
 8:         reclen  := 128,         /* 1 record = 128 byte  */
 9:
10:         putchr  := 2,           /* put console character */
11:         printf  := 9,           /* print string         */
12:         constf  := 11,          /* get console status   */
13:         openf   := 15,          /* open file    */
14:         readf   := 20,          /* read file    */
15:
16:         on      := $ff,
17:         off     := 0;
18:
19: var     fcb     at ( dfcb ),
20:         buffer  at ( dbuff ),
21:         eof, bfptr, d, i. adr[2], ch[16];
22:
23: data    crlf:   13,10,'$',
24:         nofil:  13,10,"No file$";
25:
26: {
27:         fcb[32] := 0;
28:         if bdos(openf;.fcb)=$ff  then {
29:                 bdos(printf;.nofil);
30:                 goto exit1;
31:         }
32:         eof := off; bfptr := reclen; adr[0] := adr[1] := 0;
33:         while (d:=getfile(),^eof) {
34:                 if (adr[0] & $0f) = 0 then {
35:                         bdos(printf;.crlf);
36:                         puthex(adr[1]);puthex(adr[0]);
37:                         bdos(putchr,' ');
38:                         i := -1;
39:                 }
40:                 puthex( d );
41:                 bdos(putchr,' ');
42:                 ch[inc(i)] := ?(d>=$20 & d<=$7e | d>=$a1 & d<=$df;d,'.');
43:                 ldx iy := adr; inx iy; stx adr := iy;
44:                 if (adr[0] & $0f) = 0 then {
45:                         for d:=0 to 15 {
46:                                 bdos(putchr,ch[d]);
47:                         }
48:                         if bdos(constf) then goto exit1;
49:                 }
50:         }
51: exit1:
52:         bdos(printf;.crlf);
53: }
54:
55: puthex(h);
56: {
57:         puthex1(h>>4);
58:         puthex1(h);
59: }
60:
61: puthex1(h);
62: {
63:         bdos(putchr,?((h:=h & $0f)<10;h,h+7)+'0');
64: }
65:
66:
67: getfile();
```

```
68: {
69:         if bfptr>=reclen then {
70:                 bfptr := 0;
71:                 if bdos(readf;.fcb) then {
72:                         eof := on;
73:                         return;
74:                 }
75:         }
76:         buffer[inc(bfptr)-1];
77: }
78:
79:
80: bdos(func,x;ix,#);
81: var     argn at ( _work );
82: {
83:         if argn=1 then  inline  $dd,$e5,$d1;    /* push ix    ; pop de  */
84:                   else  inline  $3a,#.x,$5f;    /* ld a,(x)   ; ld e,a  */
85:         inline  $3a,#.func,$4f;                 /* ld a,(func); ld c,a  */
86:         inline  $cd,#$0005;                     /* call 0005h           */
87:         inline  $e5,$fd,$e1;                    /* push hl    ; pop iy  */
88: }
```

●実行例2-1　ファイル・ダンプのコンパイルから実行

```
B>stellar fdump↵

  Stellar compiler Rev 1.01 ( CP/M-80,MSX-DOS Version )
  Copyright (c) 1984 H.Ohnuki / MIA

  Program  name : file_dump
  Function name : puthex
  Function name : puthexl
  Function name : getfile
  Function name : bdos

  Program  036F (0100-046E)
  Data     004B (4000-404A)

  **  End of compile,  No error(s)
B>convobj fdump↵

  Stellar utility,  convert object ==> intel HEX
  Rev 1.00  Copyright (c) 1984  H.Ohnuki / MIA

  Program  name : file_dump = 0100
  Function name : puthex = 037D
  Function name : puthexl = 03AE
                 ⦚
  Variable name : ch = 4036
  Data     name : crlf = 020D
  Data     name : nofil = 0210

  End address   : 046E
  Program size  : 036F  [ 4 Page ]
B>save 4 fdump.com↵
B>fdump fdump.com↵

  0000 C3 F7 01 C3 DF 01 C3 27 01 C3 29 01 C3 39 01 C3 ﾃ..ﾃﾟ.ﾃ'.ﾃ).ﾃ9.ﾃ
  0010 3E 01 C3 43 01 C3 44 01 C3 54 01 C3 55 01 C3 65 >.ﾃC.ﾃD.ﾃT.ﾃU.ﾃe
  0020 01 C3 66 01 C3 77 01 5E FE 5A 16 00 6A 67 3E 08 .ﾃf.ﾃw.^.Z..jg>.
  0030 29 30 01 19 3D 20 F9 7D C9 CD 43 01 7B C9 CD 44 )0..= .}ﾉﾍC.{ﾉﾍD
  0040 01 7B C9 56 5F AF 2E 08 CB 23 17 BA 38 02 92 1C .{ﾉV_ｯ..ﾋ#.ｺ8...
                 ⦚
```

●リスト2-2　　ハノイの塔

```
 1:
 2:         /*  tower of hanoi  */
 3:
 4: prog    hanoi();
 5:
 6: cons    cr := $0d,
 7:         lf := $0a;
 8: var     n;
 9: data    ms1:    "tower of hanoi",cr,lf,0,
10:         ms2:    "N ( 1 ... 9 )= ? ",0;
11: {
12:         putstr(;.ms1);
13:         {
14:             putnl(); putstr(;.ms2);
15:         } until (n:=getchr())>='1' & n<='9';
16:         n := n - '0';
17:         putnl(2);
18:         move(n,'X','Y','Z');
19: }
20:
21: %include hio.lib;
22:
23: recursive move(n,x,y,z);
24:
25: data    mvms:   "move ",0,
26:         frms:   " from ",0,
27:         toms:   " to ",0;
28: {
29:         if n>1 then {
30:             move(n-1,x,z,y);
31:             putstr(;.mvms); putchr(n+'0');
32:             putstr(;.frms); putchr(x);
33:             putstr(;.toms); putchr(y);
34:             putnl();
35:             move(n-1,z,y,x); }
36:         else {
37:             putstr(;.mvms); putchr(n+'0');
38:             putstr(;.frms); putchr(x);
39:             putstr(;.toms); putchr(y);
40:             putnl(); }
41: }
```

●リスト2-3　ハノイの塔入出力ライブラリ

```
 1:         /*********************************/
 2:         /*  tower of hanoi              */
 3:         /*                              */
 4:         /*      CP/M  console in/out    */
 5:         /*********************************/
 6:
 7:
 8: getchr();
 9: {
10:         bdos(1);
11: }
12:
13: putchr(x);
14: {
15:         bdos(2,x);
16: }
17:
18: putnl(n);
19: cons    cr := $0d,
20:         lf := $0a;
21: var     pn at ( _work );
22: {
23:         if pn=0 then n:=1;
24:         loop #,n{
25:             bdos(2,cr);
26:             bdos(2,lf);
27:         }
28: }
29:
30:
31: putstr(:ix);
32: var     x;
33: {
34:         while x:=@[ix+] {
35:             bdos(2,x);
36:         }
37: }
38:
39: bdos(func,x;ix,#);
40: var     argn at ( _work );
41: {
42:         if argn=1 then  inline  $dd,$e5,$d1;    /* push ix    ; pop de  */
43:                  else  inline  $3a,#.x,$5f;     /* ld a,(x)   ; ld e,a  */
44:         inline  $3a,#.func,$4f;                 /* ld a,(func); ld c,a  */
45:         inline  $cd,#$0005;                     /* call 0005h           */
46:         inline  $e5,$fd,$e1;                    /* push hl    ; pop iy  */
47: }
```

●実行例2-2　ハノイの塔のコンパイルから実行

```
B>stellar hanoi↵

Stellar compiler Rev 1.01 ( CP/M-80,MSX-DOS Version )
Copyright (c) 1984 H.Ohnuki / MIA

Program  name : hanoi
**  Include : hio.lib
Function name : getchr
Function name : putchr
Function name : putnl
Function name : putstr
Function name : bdos
**  End of include
Function name : move

Program  0391 (0100-0490)
Data     003A (4000-4039)

**  End of compile,  No error(s)
B>convobj hanoi↵

Stellar utility,  convert object ==> intel HEX
Rev 1.00  Copyright (c) 1984  H.Ohnuki / MIA

Program  name : hanoi = 0100
Function name : getchr = 0290
Function name : putchr = 02AF
Function name : putnl = 02D2
Function name : putstr = 031F
Function name : bdos = 0357
Function name : move = 0390
Constant name : _work = 4000
Constant name : _var = 4030
Constant name : _code = 0100
Constant name : cr = 000D
Constant name : lf = 000A
Variable name : n = 4030
Data     name : ms1 = 020D
Data     name : ms2 = 021E

End address    : 0490
Program size   : 0391  [ 4 Page ]
B>save 4 hanoi.com↵
B>hanoi↵
tower of hanoi

N ( 1 ... 9 )= ? 4

move 1 from X to Z
move 2 from X to Y
move 1 from Z to Y
move 3 from X to Z
move 1 from Y to X
move 2 from Y to Z
move 1 from X to Z
move 4 from X to Y
move 1 from Z to Y
move 2 from Z to X
move 1 from Y to X
move 3 from Z to Y
move 1 from X to Z
move 2 from X to Y
move 1 from Z to Y
```

2-3 全プログラム・リスト

ここでは，STELLAR.COM と CONVOBJ.COM の全プログラム・リストを公開します。

〔1〕使用アセンブラ

ここで使用するアセンブラは，CP／M上で動作する米マイクロソフト製のMACRO-80　V3.44です。他のアセンブラやV3.44より前のバージョンではアセンブルできないので注意してください。

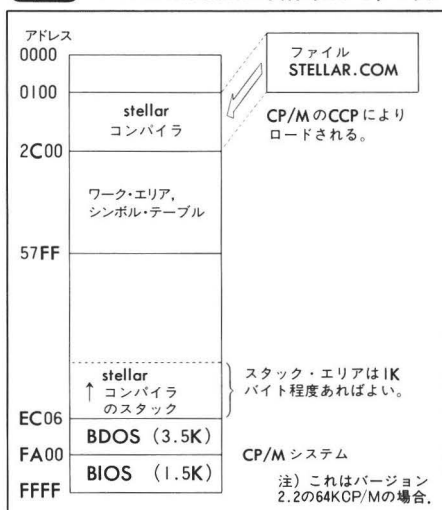
〔2〕STELLAR.COM

Stellar コンパイラは次の三つのソース・プログラムからなっています。

① **STECPM.MAC**　…入出力など，システムに依存する処理の部分。CP／M専用（移植するときは新たにつくり直す）。
② **STECMP.MAC**　…Stellar コンパイラの本体。コンパイルを行う。システムに依存しない（移植するときでも修正する必要がない）。
③ **STRUNTIM.MAC**　…ランタイム・ルーチン。オブジェクト・プログラムの一部として出力される。システムに依存しない。

これら三つのソース・プログラムは次の手順でアセンブル，リンクされ，一つのコマンド・ファイルとなります。図2-2にこのようにしてつくられたSTELLAR.COMのファイルの構成を示します。

図2-2 STELLAR.COMファイルの構成

図2-3 STELLAR.COMの実行時のメモリ・マップ

```
M80 = STECPM
M80 = STECMP
M80 = STRUNTIM
L 80/P :100/D :2C00, STECPM, STECMP, STRUNTIM, /E
SAVE 43 STELLAR.COM
```

STELLAR.COMは実行時，図2－3のようにメモリを使います。

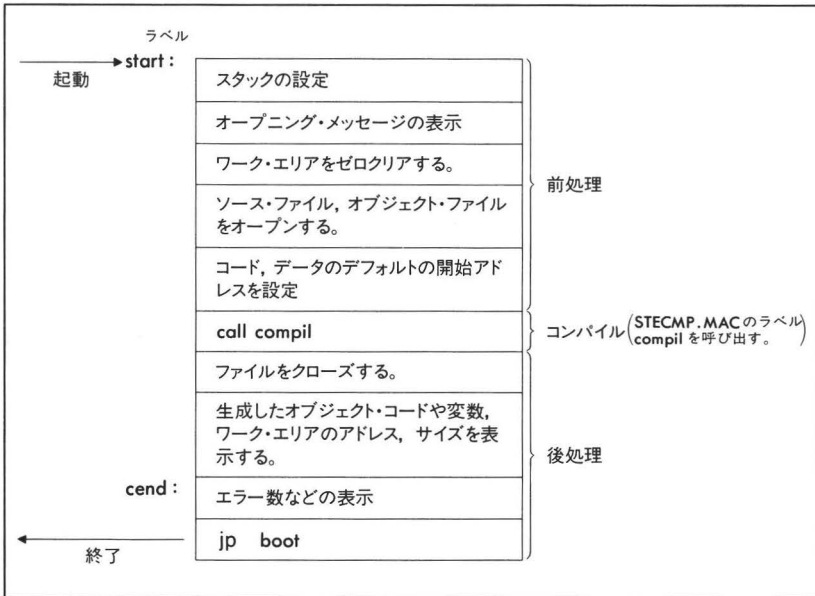
(1)STECPM.MACの構成とアセンブル・リスト

このプログラムは「起動と終了に関する部分」と「システムに依存する処理を行う部分」の二つに分けられます。

①起動と終了に関する部分

起動と終了に関する部分は，メイン・ルーチンとそれに付随するサブルーチンからなります。

メイン・ルーチンはラベル **start** から始まり，**図2－4** のような構成をしています。ここではコンパイルの前処

図2-4 メイン・ルーチンの構成

理と後処理を行っています。

メイン・ルーチンに付随する主なサブルーチンには次のようなものがあります。

fuchk：レジスタHLが示す11文字がファイル名として正しいかどうかをチェックする。

prs2ad：コードあるいはデータのサイズ，アドレスを表示する。パラメータと表示される内容は次の通り。

①レジスタDEが示すメモリに格納されている文字列。文字列は'＄'の前の文字まで表示される。

②サイズ。BC－HLの結果。

③開始アドレス。レジスタHLの値。

④終了アドレス。レジスタBCの値。

prhex4：レジスタ HL の値を16進数 4 桁で表示する。

prhex2：レジスタ A の値を16進数 2 桁で表示する。

prdec5：レジスタ HL の値を符号なし10進整数で表示する。このとき，レジスタ A の値で表示のしかたを指定する。指定する値は次のとおり。

A＝0 なら，ゼロ抑制なしで 5 桁固定長。

A＝2 なら，ゼロ抑制ありで 5 桁固定長。

A＝3 なら，ゼロ抑制ありで 1 ～ 5 桁の可変長。

②システムに依存する処理を行う部分

ここに収められたシステムに依存する処理は，STECMP からサブルーチンとして呼び出されます。他のパソコンに Stellar コンパイラを移植するときは，この部分を修正します。STECMP 側から呼び出されるサブルーチンには次のようなものがあります。

getsou：ソース・プログラムの読み込み。一回の呼び出しで 1 行分読み込む。読み込み後，レジスタやフラグに次のような値を設定する。

●レジスタ HL＝読み込んだ 1 行分の文字列の先頭アドレス。文字列の最後には NUL コード(00H）が入っている。

●レジスタ DE＝行番号（符号なし 2 進数)。

●キャリーフラグ(CY)＝0 のときリード Ok，1 のとき End of File 。

putobj：オブジェクト・ファイルへレジスタ A の値を出力する。

prtmsg：レジスタ HL が示す文字列 (NUL コードで終わる）を NUL コードの前の文字まで表示する。

error：レジスタ HL が示す文字列をエラーメッセージとして表示する。次の 3 文字が特別な意味をもつ以外は prtmsg と同じ。

●#　　文字#は現在の行番号を表示する。

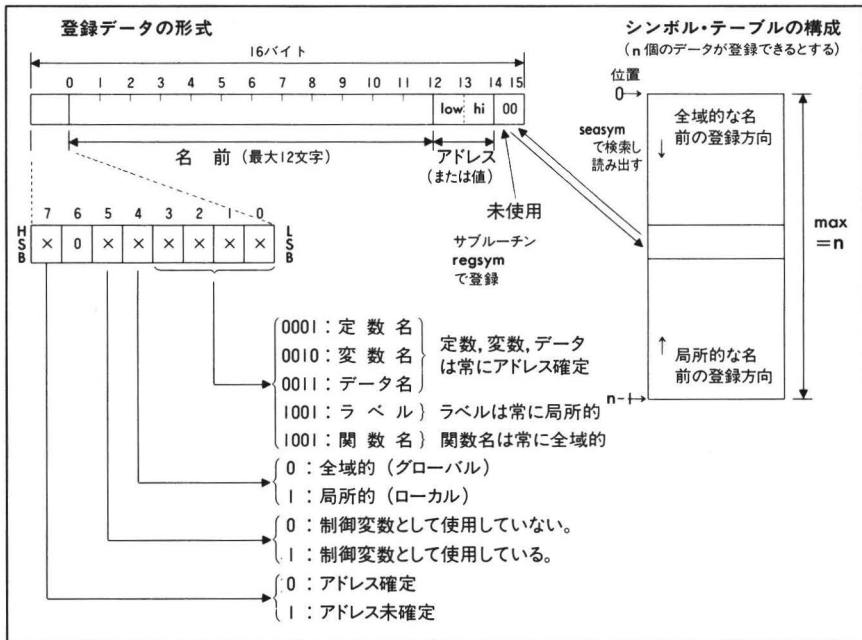
●@XX　文字@の次の 2 バイトが示す文字列

(NUL コードで終わる) を表示する。

●¥　　文字¥はレジスタ DE が示す文字列 (NUL コードで終わる) を表示する。

abort：コンパイルを中断する。

regsym：レジスタ HL が示すデータ (名前) を記号表へ登録 (再登録も含む) する。登録するデータは**図 2-5** のような形式。

図2-5 登録データの形式とシンボル・テーブル(記号表)の構成

seasym：レジスタ HL が示す名前と同じデータを記号表から検索し, 等しいものがあればレジスタ HL が示す位置へ読み出す。データの形式は登録したときと同じ。検索結果はレジスタ A に設定される。A = 0 で Ok, A =FF H で等しい名前がなかったことを示す。

elisym：シンボル・テーブル内の局所的な名前を削除する。

clrstb：シンボル・テーブルの内容をクリアする。

putglo：シンボル・テーブル内の全域的な定数名，変数名，データ名を，値あるいはアドレスと共にオブジェクト・ファイルへ出力する。

inclu：% include で指定されたファイルをオープンし，getsou が include ファイルからソース・プログラムを読み込むようにする。

edincl：getsou が **End of File** を検出したときに **STECMP** から呼び出される。**End of File** を検出したファイルが include ファイルだったら，getsou がもとのファイルからソース・プログラムを読み込むようにし，キャリーフラグを１にして戻る。include ファイルでないときはキャリーフラグを０にして戻る。

pbreak：デバッグ・モードのときコンパイル制御文% break のオブジェクト・コードを生成する（ただし CP／Mバージョンではサポートしていないので警告を表示して戻る)。

tron：デバッグ・モードのときトレース・フラグを ON にする（CP／Mバージョンではサポートしていないので警告を表示して戻る)。

troff：デバッグ・モードのときトレース・フラグを OFF にする（CP／Mバージョンではサポートしていないので警告を表示して戻る)。

ddspli：トレース・フラグが ON のとき，レジスタ **HL** に設定されている行番号（符号なし２進数）をディスプレイに表示するようなオブジェクト・コードを生成する（CP／Mバージョンではサポートしていないので何もしない)。

ddspfn：トレース・フラグが ON のとき，レジスタ **HL** が示す関数名（NUL コードで終わる文字列）を

ディスプレイに表示するようなオブジェクト・コードを生成する（CP／Mバージョンではサポートしていないので何も行わない）。

以上はSTECMPで使うサブルーチンの説明でしたが，STECMPでは次の四つの外部シンボルも使います。これらの定義もこのSTECPMで行います。

defsta：オブジェクト・プログラムのスタック・ボトムのアドレス（ただし％Ｓ：の指定がない場合）。

dsaind：スタック設定に関するオブジェクト・コードの指定。

●dsaind＝０の場合，LD SP, defsta というコードを生成する。

●dsaind＝FF Hの場合，LD SP, (defsta)というコードを生成する。

stptyp：オブジェクト・プログラム実行終了の形式指定。

●stptyp＝０の場合，HALT 命令を実行。

●stptyp＝１の場合，起動時のスタックに戻し RET 命令を実行。

●stptyp＝２の場合，次の retadr が示すアドレスへジャンプする。

retadr：stptyp＝２のとき有効となる。

☆アセンブル・リストを見る上での注意

１．各プログラムはコード相対モード，データ相対モードでアセンブルされているので，アドレスはすべて相対アドレスとなっている。

２．アドレスなどの２バイト長の値は必ず上位バイト，下位バイトの順に表示される。実際に生成される値は下位バイト，上位バイトの順になっている。

３．アドレスなどの２バイト長の値の後についている'や"，＊などの文字は次のような意味を持っている。

HHHH'　　コード相対モードのアドレスまたは値を示す。
HHHH″　　データ相対モードのアドレスまたは値を示す。
HHHH ＊　　外部参照記号を示す。

●リスト2-4　STECPMのアセンブル・リスト

```
                        title   Stellar compiler  CP/M-80,MSX-DOS support routine
                        subttl  Rev 1.01  08/30/1984
                        name    ('cpmmod')
                .z80
                ;**************************************************
                ;*                                                *
                ;*        Stellar compiler    in/out module       *
                ;*                                                *
                ;*      CP/M-80 , MSX-DOS  support routine        *
                ;*          ( Rev 1.01 )                          *
                ;*                                                *
                ;*      Copyright (c) 1984 H.Ohnuki / MIA         *
                ;*                                                *
                ;**************************************************

0101            rev     equ     101h    ;revision 1.01

                ;-------------------------------------------------
                ;-               external symbol                 -
                ;-------------------------------------------------

                        external        optisw,lino,strbuf,cptr
                        external        codorg,datorg,stkbot,cloc,dloc
                        external        work.e

                        external        compil,ptloc,ptname,synerr,gtoken
                        external        spskip,traupc

                ;-------------------------------------------------
                ;-               constant define                 -
                ;-------------------------------------------------

0100            defcda  equu    0100h   ;default code segment origin ( 0100h )
4000            defdta  equ     4000h   ;default data segment origin ( 4000h )

0006            defsta  equ     0006h   ;default stack bottom address
00FF            dsaind  equ     0ffh    ;sp set address ( 00h=direct, 0ffh=indirect )

0002            stptyp  equ     2       ;return type 2  : jump
0000            retadr  equ     0000h   ;return address : 0000h

0200            symmax  equ     512     ;max 512 symbol
0010            symlen  equ     16      ;1 symbol length = 16 byte
000C            idelen  equ     12      ;identifier length = 12 byte

000D            cr      equ     0dh     ;carriage return
000A            lf      equ     0ah     ;line feed
001A            eof     equ     1ah     ;end of file

                ;
                ;       cp/m-80,msx-dos interface
```

```
                                ;
0080                            reclen  equ     128     ; 1 record = 128 byte

0000                            boot    equ     0000h   ;system reboot entry
0005                            bdos    equ     0005h   ;bdos entry
005C                            dfcb1   equ     005ch   ;default fcb_1
006C                            dfcb2   equ     006ch   ;default fcb_2

0002                            putchrf equ     2       ;put console character
0009                            printf  equ     9       ;print string function
000F                            openf   equ     15      ;open file function
0010                            closef  equ     16      ;close file function
0013                            deletf  equ     19      ;delete file function
0014                            readf   equ     20      ;read next record
0015                            writef  equ     21      ;write next record
0016                            makef   equ     22      ;make file
001A                            setdmf  equ     26      ;set dma function

                                ;-------------------------------------------
                                ;-              global symbol              -
                                ;-------------------------------------------

                                        global          defsta,dsaind,stptyp,retadr

                                ;-------------------------------------------
                                ;-              p r o g r a m              -
                                ;-------------------------------------------

0000'                                           cseg
                                ;
                                ;       ** compile **
                                ;
0000'                           start::
0000'   2A 0006                         ld      hl,(bdos+1)
0003'   2E 00                           ld      l,0
0005'   F9                              ld      sp,hl           ;set sp reg
0006'   11 0157'                        ld      de,stamsg
0009'   0E 09                           ld      c,printf
000B'   CD 0005                         call    bdos            ;print starting message
                                ;
000E'   21 0000"                        ld      hl,work.b       ;work zero clear
0011'   36 00                           ld      (hl),0
0013'   11 0001"                        ld      de,work.b+1
0016'   01 FFFF*                        ld      bc,work.e-work.b-1
0019'   ED B0                           ldir
                                ;
001B'   3A 005D                         ld      a,(dfcb1+1)
001E'   FE 20                           cp      ' '             ;source file name ok ?
0020'   CA 024C'                        jp      z,nosou
0023'   21 0065                         ld      hl,dfcb1+9
0026'   7E                              ld      a,(hl)
0027'   FE 20                           cp      ' '
0029'   20 08                           jr      nz,stdftp1
002B'   36 53                           ld      (hl),'S'
002D'   23                              inc     hl
002E'   36 4F                           ld      (hl),'O'
0030'   23                              inc     hl
0031'   36 55                           ld      (hl),'U'        ;set file type '.SOU'
0033'                           stdftp1:
0033'   01 0009                         ld      bc,9
0036'   21 005C                         ld      hl,dfcb1
0039'   11 006C                         ld      de,dfcb2
003C'   1A                              ld      a,(de)
003D'   B7                              or      a
003E'   28 03                           jr      z,stdfdr
0040'   13                              inc     de
0041'   23                              inc     hl
0042'   0B                              dec     bc
```

```
0043'                   stdfdr:
0043'   3A 006D                 ld      a,(dfcb2+1)
0046'   FE 20                   cp      ' '
0048'   20 02                   jr      nz,stdftp2
004A'   ED B0                   ldir                    ;move dfcb1 to dfcb2
004C'                   stdftp2:
004C'   21 0075                 ld      hl,dfcb2+9
004F'   7E                      ld      a,(hl)
0050'   FE 20                   cp      ' '
0052'   20 08                   jr      nz,mvofcb
0054'   36 4F                   ld      (hl),'O'
0056'   23                      inc     hl
0057'   36 42                   ld      (hl),'B'
0059'   23                      inc     hl
005A'   36 4A                   ld      (hl),'J'        ;set file type '.OBJ'
005C'                   mvofcb:
005C'   21 006C                 ld      hl,dfcb2
005F'   11 0019"                ld      de,ofcb
0062'   01 0010                 ld      bc,16
0065'   ED B0                   ldir                    ;move dfcb2 to ofcb
                        ;
0067'   21 005D                 ld      hl,dfcb1+1
006A'   CD 014B'                call    fnchk           ;file name check
006D'   AF                      xor     a
006E'   32 007C                 ld      (dfcb1+32),a    ;reset dfcb1.cr
0071'   11 005C                 ld      de,dfcb1
0074'   0E 0F                   ld      c,openf
0076'   CD 0005                 call    bdos            ;open source file
0079'   3C                      inc     a               ;ok ?
007A'   CA 024C'                jp      z,nosou
007D'   21 0200                 ld      hl,sbfsiz
0080'   22 000B"                ld      (sbrptr),hl
0083'   22 000D"                ld      (sbrlen),hl
                        ;
0086'   21 001A"                ld      hl,ofcb+1
0089'   CD 014B'                call    fnchk           ;file name check
008C'   11 0019"                ld      de,ofcb
008F'   D5                      push    de
0090'   0E 13                   ld      c,deletf
0092'   CD 0005                 call    bdos            ;delete old object file
0095'   D1                      pop     de
0096'   0E 16                   ld      c,makef
0098'   CD 0005                 call    bdos            ;make new object file
009B'   3C                      inc     a
009C'   CA 0264'                jp      z,dirful
009F'   AF                      xor     a
00A0'   32 0039"                ld      (ofcb+32),a     ;reset ofcb.cr
00A3'   67                      ld      h,a
00A4'   6F                      ld      l,a
00A5'   22 0013"                ld      (obsptr),hl
00A8'   22 0000"                ld      (errcou),hl     ;clear error counter
                        ;
00AB'   23                      inc     hl
00AC'   22 0002"                ld      (slino),hl
00AF'   21 0100                 ld      hl,defcda
00B2'   22 0000*                ld      (codorg),hl     ;set default code segment origin
00B5'   21 4000                 ld      hl,defdta
00B8'   22 0000*                ld      (datorg),hl     ;set default data segment origin
00BB'   CD 0000*                call    compil          ;compile
                        ;
00BE'   2A 0013"                ld      hl,(obsptr)
00C1'   54                      ld      d,h
00C2'   5D                      ld      e,l
00C3'   01 055D"                ld      bc,objobf
00C6'   09                      add     hl,bc
00C7'                   objcls1:
00C7'   7B                      ld      a,e
00C8'   E6 7F                   and     reclen-1
00CA'   28 06                   jr      z,objcls2
```

```
00CC'   36 FF                   ld      (hl),0ffh       ; 0ffh = end mark
00CE'   13                      inc     de
00CF'   23                      inc     hl
00D0'   18 F5                   jr      objcls1
00D2'                   objcls2:
00D2'   EB                      ex      de,hl
00D3'   CD 0713'                call    wrobuf          ;write object buffer
00D6'   11 0019"                ld      de,ofcb
00D9'   0E 10                   ld      c,closef
00DB'   CD 0005                 call    bdos            ;close object file
00DE'   3C                      inc     a
00DF'   CA 0293'                jp      z,nocls
                        ;
00E2'   11 01B1'                ld      de,crlf
00E5'   0E 09                   ld      c,printf
00E7'   CD 0005                 call    bdos
00EA'   11 01B4'                ld      de,strpro
00ED'   ED 4B 0000*             ld      bc,(cloc)
00F1'   2A 0000*                ld      hl,(codorg)
00F4'   CD 020D'                call    prszad          ;print code size,address
00F7'   11 01C0'                ld      de,strdat
00FA'   ED 4B 0000*             ld      bc,(dloc)
00FE'   2A 0000*                ld      hl,(datorg)
0101'   CD 020D'                call    prszad          ;print data size,address
0104'   3A 0000*                ld      a,(optisw)
0107'   CB 6F                   bit     5,a             ;%s off ?
0109'   28 17                   jr      z,cend
010B'   11 01CC'                ld      de,strsta1
010E'   0E 09                   ld      c,printf
0110'   CD 0005                 call    bdos
0113'   2A 0000*                ld      hl,(stkbot)
0116'   2B                      dec     hl
0117'   CD 0308'                call    prhex4          ;print stack bottom address
011A'   11 01E3'                ld      de,strsta2
011D'   0E 09                   ld      c,printf
011F'   CD 0005                 call    bdos
0122'                   cend:
0122'   11 01E5'                ld      de,edmsg
0125'   0E 09                   ld      c,printf
0127'   CD 0005                 call    bdos
012A'   2A 0000"                ld      hl,(errcou)
012D'   7C                      ld      a,h
012E'   B5                      or      l               ;no error ?
012F'   20 0A                   jr      nz,cend1
0131'   11 01FE'                ld      de,noerr
0134'   0E 09                   ld      c,printf
0136'   CD 0005                 call    bdos            ;print "No error(s)"
0139'   18 05                   jr      cend2
013B'                   cend1:
013B'   3E 03                   ld      a,3
013D'   CD 0328'                call    prdec5          ;print error count
0140'                   cend2:
0140'   11 0201'                ld      de,errcum
0143'   0E 09                   ld      c,printf
0145'   CD 0005                 call    bdos
0148'   C3 0000                 jp      boot            ;return to os
                        ;
                        ;       **  file name check  **
                        ;
014B'                   fnchk:: ;       hl: fcb.f1 address
014B'   06 0B                   ld      b,8+3
014D'                   fnchk1:
014D'   7E                      ld      a,(hl)
014E'   FE 3F                   cp      '?'
0150'   CA 02C2'                jp      z,badfil
0153'   23                      inc     hl
0154'   10 F7                   djnz    fnchk1
0156'   C9                      ret
                        ;
```

```
                                ;       message
                                ;
0157'   0D 0A 53 74     stamsg: defb    cr,lf,'Stellar compiler Rev '
015B'   65 6C 6C 61
015F'   72 20 63 6F
0163'   6D 70 69 6C
0167'   65 72 20 52
016B'   65 76 20
016E'   31 2E 30 31             defb    rev/100h+'0','.',(rev/10h and 0fh)+'0',(rev and 0fh)+'0'
0172'   20 28 20 43             defb    ' ( CP/M-80,MSX-DOS Version )'
0176'   50 2F 4D 2D
017A'   38 30 2C 4D
017E'   53 58 2D 44
0182'   4F 53 20 56
0186'   65 72 73 69
018A'   6F 6E 20 29
018E'   0D 0A 43 6F             defb    cr,lf,'Copyright (c) 1984 H.Ohnuki / MIA'
0192'   70 79 72 69
0196'   67 68 74 20
019A'   28 63 29 20
019E'   31 39 38 34
01A2'   20 48 2E 4F
01A6'   68 6E 75 6B
01AA'   69 20 2F 20
01AE'   4D 49 41
01B1'   0D 0A 24        crlf:   defb    cr,lf,'$'
01B4'   0D 0A 50 72     strpro: defb    cr,lf,'Program  $'
01B8'   6F 67 72 61
01BC'   6D 20 20 24
01C0'   0D 0A 44 61     strdat: defb    cr,lf,'Data     $'
01C4'   74 61 20 20
01C8'   20 20 20 24
01CC'   0D 0A 53 74     strsta1:defb    cr,lf,'Stack bottom (    -$'
01D0'   61 63 6B 20
01D4'   62 6F 74 74
01D8'   6F 6D 20 20
01DC'   28 20 20 20
01E0'   20 2D 24
01E3'   29 24           strsta2:defb    ')$'
01E5'   0D 0A 0A 2A     edmsg:  defb    cr,lf,lf,'**  End of compile,  $'
01E9'   2A 20 20 45
01ED'   6E 64 20 6F
01F1'   66 20 63 6F
01F5'   6D 70 69 6C
01F9'   65 2C 20 20
01FD'   24
01FE'   4E 6F 24        noerr:  defb    'No$'
0201'   20 65 72 72     errcum: defb    ' error(s)',cr,lf,'$'
0205'   6F 72 28 73
0209'   29 0D 0A 24

                                ;
                                ;       print  size,address
                                ;
020D'                   prszad:
020D'   C5                      push    bc
020E'   E5                      push    hl
020F'   0E 09                   ld      c,printf
0211'   CD 0005                 call    bdos            ;print string
0214'   D1                      pop     de
0215'   E1                      pop     hl
0216'   E5                      push    hl
0217'   D5                      push    de
0218'   B7                      or      a
0219'   ED 52                   sbc     hl,de           ; hl = size ( byte )
021B'   7C                      ld      a,h
021C'   B5                      or      l
021D'   F5                      push    af
021E'   CD 0308'                call    prhex4
```

```
0221'   F1                      pop     af
0222'   28 25                   jr      z,prszadl
0224'   1E 20                   ld      e,' '
0226'   0E 02                   ld      c,putchrf
0228'   CD 0005                 call    bdos
022B'   1E 28                   ld      e,'('
022D'   0E 02                   ld      c,putchrf
022F'   CD 0005                 call    bdos
0232'   E1                      pop     hl
0233'   CD 0308'                call    prhex4
0236'   1E 2D                   ld      e,'-'
0238'   0E 02                   ld      c,putchrf
023A'   CD 0005                 call    bdos
023D'   E1                      pop     hl
023E'   2B                      dec     hl
023F'   CD 0308'                call    prhex4
0242'   1E 29                   ld      e,')'
0244'   0E 02                   ld      c,putchrf
0246'   C3 0005                 jp      bdos
0249'                   prszadl:
0249'   F1                      pop     af
024A'   F1                      pop     af              ; sp := sp +4
024B'   C9                      ret
                        ;
                        ;       i/o error
                        ;
024C'                   nosou:
024C'   11 0252'                ld      de,$+6
024F'   C3 02F8'                jp      errprt
0252'   0D 0A 25 4E             defb    cr,lf,'%No source file$'
0256'   6F 20 73 6F
025A'   75 72 63 65
025E'   20 66 69 6C
0262'   65 24
0264'                   dirful:
0264'   11 026A'                ld      de,$+6
0267'   C3 02F8'                jp      errprt
026A'   0D 0A 25 4E             defb    cr,lf,'%No directory space$'
026E'   6F 20 64 69
0272'   72 65 63 74
0276'   6F 72 79 20
027A'   73 70 61 63
027E'   65 24
0280'                   dskful:
0280'   11 0286'                ld      de,$+6
0283'   C3 02F8'                jp      errprt
0286'   0D 0A 25 44             defb    cr,lf,'%Disk full$'
028A'   69 73 6B 20
028E'   66 75 6C 6C
0292'   24
0293'                   nocls:
0293'   11 0299'                ld      de,$+6
0296'   C3 02F8'                jp      errprt
0299'   0D 0A 25 43             defb    cr,lf,'%Cannot close$'
029D'   61 6E 6E 6F
02A1'   74 20 63 6C
02A5'   6F 73 65 24
02A9'                   badsou:
02A9'   11 02AF'                ld      de,$+6
02AC'   C3 02F8'                jp      errprt
02AF'   0D 0A 25 42             defb    cr,lf,'%Bad source file$'
02B3'   61 64 20 73
02B7'   6F 75 72 63
02BB'   65 20 66 69
02BF'   6C 65 24
02C2'                   badfil:
02C2'   11 02C8'                ld      de,$+6
02C5'   C3 02F8'                jp      errprt
02C8'   0D 0A 25 42             defb    cr,lf,'%Bad file name$'
```

```
02CC'   61 64 20 66
02D0'   69 6C 65 20
02D4'   6E 61 6D 65
02D8'   24
02D9'                   sytovf:
02D9'   11 02DF'                ld      de,$+6
02DC'   C3 02F8'                jp      errprt
02DF'   0D 0A 25 53             defb    cr,lf,'%Symbol table overflow$'
02E3'   79 6D 62 6F
02E7'   6C 20 74 61
02EB'   62 6C 65 20
02EF'   6F 76 65 72
02F3'   66 6C 6F 77
02F7'   24
                        ;
02F8'                   errprt:
02F8'   0E 09                   ld      c,printf
02FA'   CD 0005                 call    bdos
02FD'   11 01B1'                ld      de,crlf
0300'   0E 09                   ld      c,printf
0302'   CD 0005                 call    bdos
0305'   C3 0000                 jp      boot            ;return to os
                        ;
                        ;       **  print hex  **
                        ;
0308'                   prhex4:: ;      hl: output data
0308'   E5                      push    hl
0309'   7C                      ld      a,h
030A'   CD 030F'                call    prhex2
030D'   E1                      pop     hl
030E'   7D                      ld      a,l
                        ;
030F'                   prhex2::
030F'   F5                      push    af
0310'   0F                      rrca
0311'   0F                      rrca
0312'   0F                      rrca
0313'   0F                      rrca
0314'   CD 0318'                call    prhex1
0317'   F1                      pop     af
                        ;
0318'                   prhex1::
0318'   E6 0F                   and     0fh
031A'   C6 30                   add     a,'0'
031C'   FE 3A                   cp      '9'+1
031E'   38 02                   jr      c,$+4
0320'   C6 07                     add   a,'A'-'9'-1
0322'   5F                      ld      e,a
0323'   0E 02                   ld      c,putchrf
0325'   C3 0005                 jp      bdos
                        ;
                        ;       **  print decimal  **
                        ;
0328'                   prdec5:: ;      a : 00h=no zero suppression
                                 ;          02h=zero suppression, fixed print   ( 5 digit )
                                 ;          03h=zero suppression, varying print ( 1...5 digit )
                                 ;      hl: out put data
0328'   57                      ld      d,a
0329'   01 2710                 ld      bc,10000
032C'   CD 034A'                call    prdecs
032F'   01 03E8                 ld      bc,1000
0332'   CD 034A'                call    prdecs
0335'   01 0064                 ld      bc,100
0338'   CD 034A'                call    prdecs
033B'   01 000A                 ld      bc,10
033E'   CD 034A'                call    prdecs
0341'   7D                      ld      a,l
0342'   F6 30                   or      '0'
0344'   5F                      ld      e,a
```

```
0345'   0E 02                   ld      c,putchrf
0347'   C3 0005                 jp      bdos
                        ;
034A'                   prdecs:
034A'   1E 30                   ld      e,'0'
034C'                   prdecs1:
034C'   B7                      or      a
034D'   ED 42                   sbc     hl,bc
034F'   1C                      inc     e
0350'   30 FA                   jr      nc,prdecs1
0352'   09                      add     hl,bc
0353'   1D                      dec     e
0354'   CB 4A                   bit     1,d
0356'   28 0D                   jr      z,prdecs3
0358'   7B                      ld      a,e
0359'   FE 30                   cp      '0'
035B'   20 06                   jr      nz,prdecs2
035D'   CB 42                   bit     0,d
035F'   C0                      ret     nz
0360'   1E 20                   ld      e,' '
0362'   01                      defb    01h
0363'                   prdecs2:
0363'   CB 8A                   res     1,d
0365'                   prdecs3:
0365'   E5                      push    hl
0366'   D5                      push    de
0367'   0E 02                   ld      c,putchrf
0369'   CD 0005                 call    bdos
036C'   D1                      pop     de
036D'   E1                      pop     hl
036E'   C9                      ret
                        ;
                        ;       **  abort  **
                        ;
036F'                   abort::
036F'   11 037A'                ld      de,abtms
0372'   0E 09                   ld      c,printf
0374'   CD 0005                 call    bdos
0377'   C3 0122'                jp      cend
                        ;
037A'   0D 0A 25 41     abtms:  defb    cr,lf,'%Abort$'
037E'   62 6F 72 74
0382'   24
                        ;
                        ;       **  error  **
                        ;
0383'                   error:: ;       hl: error message address (nul end string )
                                ;       de: sub message address   ( nul end string )
0383'   ED 53 005B"             ld      (sbmsad),de
0387'   2B                      dec     hl
0388'   E5                      push    hl
0389'   11 01B1'                ld      de,crlf
038C'   0E 09                   ld      c,printf
038E'   CD 0005                 call    bdos
0391'                   error1:
0391'   E1                      pop     hl
0392'   23                      inc     hl
0393'   7E                      ld      a,(hl)
0394'   B7                      or      a
0395'   28 3E                   jr      z,error5
0397'   E5                      push    hl
0398'   FE 23                   cp      '#'
039A'   20 15                   jr      nz,error2
039C'   2A 0000*                ld      hl,(lino)
039F'   3E 02                   ld      a,2
03A1'   CD 0328'                call    prdec5          ;print line number
03A4'   11 03AE'                ld      de,error1.5
03A7'   0E 09                   ld      c,printf
03A9'   CD 0005                 call    bdos
```

```
03AC'   18 E3                 jr      error1
03AE'   3A 20 24      error1.5:       defb    ': $'
03B1'                 error2:
03B1'   FE 40                 cp      '@'
03B3'   20 0C                 jr      nz,error3
03B5'   E1                    pop     hl
03B6'   23                    inc     hl
03B7'   5E                    ld      e,(hl)
03B8'   23                    inc     hl
03B9'   56                    ld      d,(hl)
03BA'   E5                    push    hl
03BB'   EB                    ex      de,hl
03BC'   CD 03F0'              call    prtmsg          ;print sub message
03BF'   18 D0                 jr      error1
03C1'                 error3:
03C1'   FE 5C                 cp      '¥'
03C3'   20 08                 jr      nz,error4
03C5'   2A 005B"              ld      hl,(sbmsad)
03C8'   CD 03F0'              call    prtmsg          ;print sub message
03CB'   18 C4                 jr      error1
03CD'                 error4:
03CD'   5F                    ld      e,a
03CE'   0E 02                 ld      c,putchrf
03D0'   CD 0005               call    bdos
03D3'   18 BC                 jr      error1
                      ;
03D5'                 error5:
03D5'   2A 0000"              ld      hl,(errcou)
03D8'   23                    inc     hl
03D9'   22 0000"              ld      (errcou),hl
03DC'   11 01B1'              ld      de,crlf
03DF'   0E 09                 ld      c,printf
03E1'   CD 0005               call    bdos
03E4'   21 005D"              ld      hl,linbuf
03E7'   3A 000A"              ld      a,(inclf)
03EA'   B7                    or      a
03EB'   28 03                 jr      z,$+5
03ED'   21 00DD"                ld    hl,ilibuf
                      ;
                      ;       **  print message  **
                      ;
03F0'                 prtmsg:: ;      hl: string address ( nul code end )
03F0'   7E                    ld      a,(hl)
03F1'   B7                    or      a
03F2'   C8                    ret     z
03F3'   E5                    push    hl
03F4'   5F                    ld      e,a
03F5'   0E 02                 ld      c,putchrf
03F7'   CD 0005               call    bdos
03FA'   E1                    pop     hl
03FB'   23                    inc     hl
03FC'   18 F2                 jr      prtmsg
                      ;
                      ;       **  make fcb  **
                      ;
03FE'                 mkfcb:: ;       de: fcb address
                              ;       hl: file name
03FE'   CD 0000*              call    spskip          ;space skip
0401'   EB                    ex      de,hl
0402'   1A                    ld      a,(de)
0403'   B7                    or      a
0404'   28 0D                 jr      z,mkfcb1
0406'   CD 0000*              call    traupc          ;translate to upper case
0409'   D6 40                 sub     '@'
040B'   4F                    ld      c,a
040C'   13                    inc     de
040D'   1A                    ld      a,(de)
040E'   FE 3A                 cp      ':'
0410'   28 05                 jr      z,mkfcb2
```

```
0412'   1B                      dec     de
0413'                   mkfcb1:
0413'   36 00                   ld .    (hl),0
0415'   18 02                   jr      mkfcb3
0417'                   mkfcb2:
0417'   71                      ld      (hl),c          ;set fcb.dr
0418'   13                      inc     de
0419'                   mkfcb3:
0419'   06 08                   ld      b,8             ;set fcb.f1 ... fcb.f8
041B'   23                      inc     hl
041C'                   mkfcb4:
041C'   CD 0460'                call    chckfc          ;character check
041F'   28 0D                   jr      z,mkfcb6
0421'   77                      ld      (hl),a
0422'   13                      inc     de
0423'   23                      inc     hl
0424'   10 F6                   djnz    mkfcb4
0426'                   mkfcb5:
0426'   CD 0460'                call    chckfc
0429'   28 08                   jr      z,mkfcb7
042B'   13                      inc     de
042C'   18 F8                   jr      mkfcb5
042E'                   mkfcb6:
042E'   36 20                   ld      (hl),' '
0430'   23                      inc     hl
0431'   10 FB                   djnz    mkfcb6
0433'                   mkfcb7:
0433'   06 03                   ld      b,3             ;set fcb.t1 ... fcb.t3
0435'   FE 2E                   cp      '.'
0437'   20 13                   jr      nz,mkfcb10
0439'   13                      inc     de
043A'                   mkfcb8:
043A'   CD 0460'                call    chckfc          ;character check
043D'   28 0D                   jr      z,mkfcb10
043F'   77                      ld      (hl),a
0440'   13                      inc     de
0441'   23                      inc     hl
0442'   10 F6                   djnz    mkfcb8
0444'                   mkfcb9:
0444'   CD 0460'                call    chckfc
0447'   28 08                   jr      z,mkfcb11
0449'   13                      inc     de
044A'   18 F8                   jr      mkfcb9
044C'                   mkfcb10:
044C'   36 20                   ld      (hl),' '
044E'   23                      inc     hl
044F'   10 FB                   djnz    mkfcb10
0451'                   mkfcb11:
0451'   06 03                   ld      b,3
0453'   AF                      xor     a               ;clear fcb.ex, s1, s2
0454'                   mkfcb12:
0454'   77                      ld      (hl),a
0455'   23                      inc     hl
0456'   10 FC                   djnz    mkfcb12
0458'   01 0011                 ld      bc,17
045B'   09                      add     hl,bc
045C'   77                      ld      (hl),a          ;clear fcb.rc
045D'   EB                      ex      de,hl
045E'   B7                      or      a               ; cy=0 : make ok !
045F'   C9                      ret
                        ;
0460'                   chckfc:
0460'   1A                      ld      a,(de)
0461'   CD 0000*                call    traupc          ;translate to upper case
0464'   B7                      or      a
0465'   C8                      ret     z
0466'   FE 20                   cp      ' '
0468'   C8                      ret     z
0469'   38 22                   jr      c,nomak
```

```
046B'   FE 2E               cp      '.'
046D'   C8                  ret     z
046E'   FE 2C               cp      ','
0470'   C8                  ret     z
0471'   FE 3A               cp      ':'
0473'   C8                  ret     z
0474'   FE 3B               cp      ';'
0476'   C8                  ret     z
0477'   FE 3C               cp      '<'
0479'   C8                  ret     z
047A'   FE 3D               cp      '='
047C'   C8                  ret     z
047D'   FE 3E               cp      '>'
047F'   C8                  ret     z
0480'   FE 5B               cp      '['
0482'   C8                  ret     z
0483'   FE 5D               cp      ']'
0485'   C8                  ret     z               ; z=1 : delimiting character
0486'   FE 3F               cp      '?'
0488'   28 03               jr      z,nomak
048A'   FE 2A               cp      '*'
048C'   C0                  ret     nz              ; z=0 : non delimiting character
048D'               nomak:
048D'   F1                  pop     af
048E'   37                  scf                     ; cy=1 : bad file name
048F'   C9                  ret
                    ;
                    ;       **  include file open  **
                    ;
0490'               inclu::
0490'   3A 000A"            ld      a,(inclf)
0493'   B7                  or      a
0494'   C2 0000*            jp      nz,synerr
0497'   2A 0000*            ld      hl,(cptr)
049A'   CD 0000*            call    spskip
049D'   E5                  push    hl
049E'   11 003A"            ld      de,ifcb
04A1'   CD 03FE'            call    mkfcb           ;make fcb
04A4'   DA 0553'            jp      c,badifl
04A7'   22 0000*            ld      (cptr),hl
04AA'   E5                  push    hl
04AB'   CD 0000*            call    gtoken
04AE'   FE 3B               cp      ';'
04B0'   C2 0000*            jp      nz,synerr
04B3'   11 003A"            ld      de,ifcb
04B6'   0E 0F               ld      c,openf
04B8'   CD 0005             call    bdos            ;file open
04BB'   3C                  inc     a
04BC'   CA 0573'            jp      z,noifl
04BF'   21 0200             ld      hl,ibfsiz
04C2'   22 000F"            ld      (ibrptr),hl
04C5'   22 0011"            ld      (ibrlen),hl
                    ;
04C8'   3E FF               ld      a,0ffh
04CA'   32 000A"            ld      (inclf),a
04CD'   2A 0000*            ld      hl,(cptr)
04D0'   22 0006"            ld      (cptr1),hl
04D3'   2A 0000*            ld      hl,(lino)
04D6'   22 0004"            ld      (lino1),hl
04D9'   2A 0002"            ld      hl,(slino)
04DC'   22 0008"            ld      (slino1),hl
04DF'   21 0001             ld      hl,1
04E2'   22 0002"            ld      (slino),hl
04E5'   21 00DD"            ld      hl,ilibuf
04E8'   22 0000*            ld      (cptr),hl
04EB'   36 00               ld      (hl),0
                    ;
04ED'   21 050A'            ld      hl,inclflm      ;print include file name
04F0'   CD 03F0'            call    prtmsg
```

```
04F3'   E1                      pop     hl
04F4'   D1                      pop     de
04F5'   B7                      or      a
04F6'   ED 52                   sbc     hl,de
04F8'   EB                      ex      de,hl
04F9'               prifnl:
04F9'   7A                      ld      a,d
04FA'   B3                      or      e
04FB'   C8                      ret     z
04FC'   D5                      push    de
04FD'   E5                      push    hl
04FE'   5E                      ld      e,(hl)
04FF'   0E 02                   ld      c,putchrf
0501'   CD 0005                 call    bdos
0504'   E1                      pop     hl
0505'   D1                      pop     de
0506'   23                      inc     hl
0507'   1B                      dec     de
0508'   18 EF                   jr      prifnl
                    ;
050A'   0D 0A 2A 2A inclflm:defb        cr,lf,'**  Include : ',0
050E'   20 20 49 6E
0512'   63 6C 75 64
0516'   65 20 3A 20
051A'   00
                    ;
                    ;       **  end of include  **
                    ;
051B'               edincl::
051B'   3A 000A"                ld      a,(inclf)
051E'   B7                      or      a
051F'   C8                      ret     z               ;cy=0 : end of source
0520'   AF                      xor     a
0521'   32 000A"                ld      (inclf),a
0524'   21 053E'                ld      hl,edicm
0527'   CD 03F0'                call    prtmsg
052A'   2A 0004"                ld      hl,(linol)
052D'   22 0000*                ld      (lino),hl
0530'   2A 0008"                ld      hl,(slinol)
0533'   22 0002"                ld      (slino),hl
0536'   2A 0006"                ld      hl,(cptrl)
0539'   22 0000*                ld      (cptr),hl
053C'   37                      scf                     ; cy=1 : end of include
053D'   C9                      ret
                    ;
053E'   0D 0A 2A 2A edicm:  defb        cr,lf,'**  End of include',0
0542'   20 20 45 6E
0546'   64 20 6F 66
054A'   20 69 6E 63
054E'   6C 75 64 65
0552'   00
                    ;
                    ;       **  include error  **
                    ;
0553'               badifl:
0553'   21 055C'                ld      hl,badiflm
0556'   CD 0383'                call    error
0559'   C3 036F'                jp      abort
055C'   23 42 61 64 badiflm:defb        '#Bad include file name',0
0560'   20 69 6E 63
0564'   6C 75 64 65
0568'   20 66 69 6C
056C'   65 20 6E 61
0570'   6D 65 00
                    ;
0573'               noifl:
0573'   21 057C'                ld      hl,noiflm
0576'   CD 0383'                call    error
0579'   C3 036F'                jp      abort
```

```
057C'   23 4E 6F 20   noiflm: defb    '#No include file',0
0580'   69 6E 63 6C
0584'   75 64 65 20
0588'   66 69 6C 65
058C'   00
                      ;
                      ;       **  break, tron, troff  **
                      ;
058D'                 pbreak::
058D'   11 0593'              ld      de,$+6
0590'   C3 05B3'              jp      nosup
0593'   25 62 72 65           defb    '%break',0
0597'   61 6B 00
059A'                 tron::
059A'   11 05A0'              ld      de,$+6
059D'   C3 05B3'              jp      nosup
05A0'   25 74 72 6F           defb    '%tron',0
05A4'   6E 00
05A6'                 troff::
05A6'   11 05AC'              ld      de,$+6
05A9'   C3 05B3'              jp      nosup
05AC'   25 74 72 6F           defb    '%troff',0
05B0'   66 66 00
05B3'                 nosup:
05B3'   3A 0000*              ld      a,(optisw)
05B6'   CB 67                 bit     4,a             ;%debug sw on ?
05B8'   C8                    ret     z
05B9'   21 05C7'              ld      hl,nosupm
05BC'   CD 0383'              call    error
05BF'   2A 0000"              ld      hl,(errcou)
05C2'   2B                    dec     hl
05C3'   22 0000"              ld      (errcou),hl
05C6'   C9                    ret
05C7'   23 4E 6F 6E   nosupm: defb    '#Non supporting (warning): ¥',0
05CB'   20 73 75 70
05CF'   70 6F 72 74
05D3'   69 6E 67 20
05D7'   28 77 61 72
05DB'   6E 69 6E 67
05DF'   29 3A 20 5C
05E3'   00
                      ;
                      ;       **  function name, line number display object out  **
                      ;
05E4'                 ddspfn::
05E4'                 ddspli::                ;non supporting
05E4'   C9                    ret
                      ;
                      ;       **  get source file  **
                      ;
05E5'                 getsou::
05E5'   21 005D"              ld      hl,linbuf
05E8'   3A 000A"              ld      a,(inclf)
05EB'   B7                    or      a
05EC'   28 03                 jr      z,$+5
05EE'   21 00DD"                ld    hl,ilibuf
05F1'   06 80                 ld      b,linbufs
05F3'   E5                    push    hl
05F4'                 getsoul:
05F4'   C5                    push    bc
05F5'   E5                    push    hl
05F6'   3A 000A"              ld      a,(inclf)
05F9'   B7                    or      a
05FA'   20 05                 jr      nz,getsoul.3
05FC'   CD 062F'              call    rdsouc          ;read source file
05FF'   18 03                 jr      getsoul.6
0601'                 getsoul.3:
0601'   CD 0692'              call    rdincl
0604'                 getsoul.6:
```

```
0604'   E1                      pop     hl
0605'   C1                      pop     bc
0606'   FE 1A                   cp      eof             ;end of file ?
0608'   37                      scf
0609'   28 0F                   jr      z,getsou2
060B'   FE 0D                   cp      cr              ;cr ?
060D'   28 13                   jr      z,getsou3
060F'   FE 0A                   cp      lf              ;lf ?
0611'   28 E1                   jr      z,getsou1
0613'   77                      ld      (hl),a
0614'   23                      inc     hl
0615'   10 DD                   djnz    getsou1
0617'   C3 02A9'                jp      badsou
061A'                   getsou2:
061A'   36 00                   ld      (hl),0
061C'   ED 5B 0002"             ld      de,(slino)
0620'   18 0B                   jr      getsou4
0622'                   getsou3:
0622'   36 00                   ld      (hl),0
0624'   2A 0002"                ld      hl,(slino)
0627'   54                      ld      d,h
0628'   5D                      ld      e,l
0629'   23                      inc     hl
062A'   22 0002"                ld      (slino),hl
062D'                   getsou4:
062D'   E1                      pop     hl
062E'   C9                      ret
                        ;
                        ;       **  read source file ( 1 character )  **
                        ;
062F'                   rdsouc::
062F'   2A 000B"                ld      hl,(sbrptr)
0632'   ED 5B 000D"             ld      de,(sbrlen)
0636'   E5                      push    hl
0637'   B7                      or      a
0638'   ED 52                   sbc     hl,de
063A'   E1                      pop     hl              ; sbrptr < sbrlen ?
063B'   38 45                   jr      c,rdsouc5
063D'   7B                      ld      a,e
063E'   FE 00                   cp      low sbfsiz
0640'   20 05                   jr      nz,rdsouc0
0642'   7A                      ld      a,d
0643'   FE 02                   cp      high sbfsiz
0645'   28 04                   jr      z,rdsouc1
0647'                   rdsouc0:
0647'   37                      scf                     ; cy=1 : end of file
0648'   3E 1A                   ld      a,eof           ; a = eof character
064A'   C9                      ret
064B'                   rdsouc1:
064B'   21 0000                 ld      hl,0
064E'   22 000B"                ld      (sbrptr),hl
0651'   11 015D"                ld      de,souibf
0654'   06 04                   ld      b,sbfsiz/reclen
0656'                   rdsouc2:
0656'   C5                      push    bc
0657'   E5                      push    hl
0658'   D5                      push    de
0659'   0E 1A                   ld      c,setdmf
065B'   CD 0005                 call    bdos            ;set dma address
065E'   11 005C                 ld      de,dfcb1
0661'   0E 14                   ld      c,readf
0663'   CD 0005                 call    bdos            ;read source file
0666'   E1                      pop     hl
0667'   01 0080                 ld      bc,reclen
066A'   09                      add     hl,bc
066B'   EB                      ex      de,hl
066C'   E1                      pop     hl
066D'   09                      add     hl,bc
066E'   B7                      or      a               ;end of file ?
```

```
066F'   C1                      pop     bc
0670'   20 04                   jr      nz,rdsouc3
0672'   10 E2                   djnz    rdsouc2
0674'   18 09                   jr      rdsouc4
0676'                   rdsouc3:
0676'   01 FF80                 ld      bc,-reclen
0679'   09                      add     hl,bc
067A'   22 000D"                ld      (sbrlen),hl
067D'   18 B0                   jr      rdsouc
067F'                   rdsouc4:
067F'   22 000D"                ld      (sbrlen),hl
0682'                   rdsouc5:
0682'   2A 000B"                ld      hl,(sbrptr)
0685'   E5                      push    hl
0686'   11 015D"                ld      de,souibf
0689'   19                      add     hl,de
068A'   7E                      ld      a,(hl)          ; a = character
068B'   E1                      pop     hl
068C'   23                      inc     hl
068D'   22 000B"                ld      (sbrptr),hl
0690'   B7                      or      a               ; cy=0 : read ok
0691'   C9                      ret
                        ;
                        ;       **  read include file ( 1 character )  **
                        ;
0692'                   rdincl::
0692'   2A 000F"                ld      hl,(ibrptr)
0695'   ED 5B 0011"             ld      de,(ibrlen)
0699'   E5                      push    hl
069A'   B7                      or      a
069B'   ED 52                   sbc     hl,de
069D'   E1                      pop     hl              ; ibrptr < ibrlen ?
069E'   38 45                   jr      c,rdincl5
06A0'   7B                      ld      a,e
06A1'   FE 00                   cp      low ibfsiz
06A3'   20 05                   jr      nz,rdincl0
06A5'   7A                      ld      a,d
06A6'   FE 02                   cp      high ibfsiz
06A8'   28 04                   jr      z,rdincl1
06AA'                   rdincl0:
06AA'   37                      scf                     ; cy=1 : end of file
06AB'   3E 1A                   ld      a,eof           ; a = eof character
06AD'   C9                      ret
06AE'                   rdincl1:
06AE'   21 0000                 ld      hl,0
06B1'   22 000F"                ld      (ibrptr),hl
06B4'   11 035D"                ld      de,incibf
06B7'   06 04                   ld      b,ibfsiz/reclen
06B9'                   rdincl2:
06B9'   C5                      push    bc
06BA'   E5                      push    hl
06BB'   D5                      push    de
06BC'   0E 1A                   ld      c,setdmf
06BE'   CD 0005                 call    bdos            ;set dma address
06C1'   11 003A"                ld      de,ifcb
06C4'   0E 14                   ld      c,readf
06C6'   CD 0005                 call    bdos            ;read source file
06C9'   E1                      pop     hl
06CA'   01 0080                 ld      bc,reclen
06CD'   09                      add     hl,bc
06CE'   EB                      ex      de,hl
06CF'   E1                      pop     hl
06D0'   09                      add     hl,bc
06D1'   B7                      or      a               ;end of file ?
06D2'   C1                      pop     bc
06D3'   20 04                   jr      nz,rdincl3
06D5'   10 E2                   djnz    rdincl2
06D7'   18 09                   jr      rdincl4
06D9'                   rdincl3:
```

```
06D9'   01 FF80             ld      bc,-reclen
06DC'   09                  add     hl,bc
06DD'   22 0011"            ld      (ibrlen),hl
06E0'   18 B0               jr      rdincl
06E2'               rdincl4:
06E2'   22 0011"            ld      (ibrlen),hl
06E5'               rdincl5:
06E5'   2A 000F"            ld      hl,(ibrptr)
06E8'   E5                  push    hl
06E9'   11 035D"            ld      de,incibf
06EC'   19                  add     hl,de
06ED'   7E                  ld      a,(hl)          ; a = character
06EE'   E1                  pop     hl
06EF'   23                  inc     hl
06F0'   22 000F"            ld      (ibrptr),hl
06F3'   B7                  or      a               ; cy=0 : read ok
06F4'   C9                  ret
                    ;
                    ;       **  put object file ( 1 byte )  **
                    ;
06F5'               putobj:: ;      a : object
06F5'   2A 0013"            ld      hl,(obsptr)
06F8'   E5                  push    hl
06F9'   11 055D"            ld      de,objobf
06FC'   19                  add     hl,de
06FD'   77                  ld      (hl),a
06FE'   E1                  pop     hl
06FF'   23                  inc     hl
0700'   22 0013"            ld      (obsptr),hl
0703'   7D                  ld      a,l
0704'   FE 00               cp      low obfsiz
0706'   C0                  ret     nz
0707'   7C                  ld      a,h
0708'   D6 04               sub     high obfsiz     ; obfptr <> obfsiz ?
070A'   C0                  ret     nz
070B'   67                  ld      h,a
070C'   6F                  ld      l,a
070D'   22 0013"            ld      (obsptr),hl
0710'   21 0400             ld      hl,obfsiz
0713'               wrobuf:
0713'   11 055D"            ld      de,objobf
0716'               wrobuf1:
0716'   E5                  push    hl
0717'   D5                  push    de
0718'   0E 1A               ld      c,setdmf
071A'   CD 0005             call    bdos            ;set dma address
071D'   11 0019"            ld      de,ofcb
0720'   0E 15               ld      c,writef
0722'   CD 0005             call    bdos            ;write object file
0725'   E1                  pop     hl
0726'   11 0080             ld      de,reclen
0729'   19                  add     hl,de
072A'   EB                  ex      de,hl
072B'   E1                  pop     hl
072C'   01 FF80             ld      bc,-reclen
072F'   09                  add     hl,bc
0730'   B7                  or      a               ;write ok ?
0731'   C2 0280'            jp      nz,dskful
0734'   7C                  ld      a,h
0735'   B5                  or      l
0736'   20 DE               jr      nz,wrobuf1
0738'   C9                  ret
                    ;
                    ;       **  symbol table register  **
                    ;
0739'               regsym:: ;      hl: register data
0739'   CB 66               bit     4,(hl)
073B'   28 2A               jr      z,regsym5
                    ;
```

```
                        ; local symbol
                        ;
073D'   11 294D"              ld      de,symtbl+(symmax-1)*symlen
0740'                   regsym1:
0740'   01 0C00               ld      bc,idelen*100h
0743'   E5                    push    hl
0744'   D5                    push    de
0745'   2A 0017"              ld      hl,(lcptr)
0748'   B7                    or      a
0749'   ED 52                 sbc     hl,de
074B'   28 44                 jr      z,regsym10
074D'   D1                    pop     de
074E'   E1                    pop     hl
074F'   E5                    push    hl
0750'   D5                    push    de
0751'                   regsym2:
0751'   23                    inc     hl
0752'   13                    inc     de
0753'   1A                    ld      a,(de)
0754'   BE                    cp      (hl)
0755'   20 07                 jr      nz,regsym3
0757'   B7                    or      a
0758'   28 5E                 jr      z,regsym13
075A'   10 F5                 djnz    regsym2
075C'   18 5A                 jr      regsym13
075E'                   regsym3:
075E'   D1                    pop     de
075F'   21 FFF0               ld      hl,-symlen
0762'   19                    add     hl,de
0763'   EB                    ex      de,hl
0764'   E1                    pop     hl
0765'   18 D9                 jr      regsym1
                        ;
                        ; global symbol
                        ;
0767'                   regsym5:
0767'   11 095D"              ld      de,symtbl
076A'                   regsym6:
076A'   01 0C00               ld      bc,idelen*100h
076D'   E5                    push    hl
076E'   D5                    push    de
076F'   2A 0015"              ld      hl,(glptr)
0772'   B7                    or      a
0773'   ED 52                 sbc     hl,de
0775'   28 26                 jr      z,regsym11
0777'   D1                    pop     de
0778'   E1                    pop     hl
0779'   E5                    push    hl
077A'   D5                    push    de
077B'                   regsym7:
077B'   23                    inc     hl
077C'   13                    inc     de
077D'   1A                    ld      a,(de)
077E'   BE                    cp      (hl)
077F'   20 07                 jr      nz,regsym8
0781'   B7                    or      a
0782'   28 34                 jr      z,regsym13
0784'   10 F5                 djnz    regsym7
0786'   18 30                 jr      regsym13
0788'                   regsym8:
0788'   D1                    pop     de
0789'   21 0010               ld      hl,symlen
078C'   19                    add     hl,de
078D'   EB                    ex      de,hl
078E'   E1                    pop     hl
078F'   18 D9                 jr      regsym6
                        ;
0791'                   regsym10:
0791'   2A 0017"              ld      hl,(lcptr)
```

```
0794'   11 FFF0              ld      de,-symlen
0797'   19                   add     hl,de
0798'   22 0017"             ld      (lcptr),hl
079B'   18 0A                jr      regsym12
079D'               regsym11:
079D'   2A 0015"             ld      hl,(glptr)
07A0'   11 0010              ld      de,symlen
07A3'   19                   add     hl,de
07A4'   22 0015"             ld      (glptr),hl
07A7'               regsym12:
07A7'   2A 0017"             ld      hl,(lcptr)
07AA'   11 0020              ld      de,symlen*2
07AD'   19                   add     hl,de
07AE'   ED 5B 0015"          ld      de,(glptr)
07B2'   B7                   or      a
07B3'   ED 52                sbc     hl,de           ;symbol table overflow ?
07B5'   CA 02D9'             jp      z,sytovf
07B8'               regsym13:
07B8'   D1                   pop     de
07B9'   E1                   pop     hl
07BA'   01 0010              ld      bc,symlen
07BD'   ED B0                ldir
07BF'   C9                   ret
                    ;
                    ;        **  symbol table search  **
                    ;
07C0'               seasym:: ;       hl: search data
07C0'   11 294D"             ld      de,symtbl+(symmax-1)*symlen
07C3'               seasym1:
07C3'   01 0C00              ld      bc,idelen*100h
07C6'   E5                   push    hl
07C7'   D5                   push    de
07C8'   1A                   ld      a,(de)
07C9'   B7                   or      a
07CA'   28 14                jr      z,seasym3
07CC'               seasym2:
07CC'   23                   inc     hl
07CD'   13                   inc     de
07CE'   1A                   ld      a,(de)
07CF'   BE                   cp      (hl)
07D0'   20 13                jr      nz,seasym4
07D2'   B7                   or      a
07D3'   28 02                jr      z,$+4
07D5'   10 F5                djnz    seasym2
07D7'   E1                   pop     hl
07D8'   D1                   pop     de
07D9'   01 0010              ld      bc,symlen
07DC'   ED B0                ldir
07DE'   AF                   xor     a
07DF'   C9                   ret
07E0'               seasym3:
07E0'   E1                   pop     hl
07E1'   2A 0015"             ld      hl,(glptr)
07E4'   FE                   defb    0feh
07E5'               seasym4:
07E5'   E1                   pop     hl
07E6'   11 FFF0              ld      de,-symlen
07E9'   19                   add     hl,de
07EA'   E5                   push    hl
07EB'   11 095D"             ld      de,symtbl
07EE'   B7                   or      a
07EF'   ED 52                sbc     hl,de
07F1'   D1                   pop     de
07F2'   E1                   pop     hl
07F3'   30 CE                jr      nc,seasym1
07F5'   F6 FF                or      0ffh
07F7'   C9                   ret
                    ;
                    ;        **  elimination local symbol  **
```

```
                        ;
07F8'                   elisym::
07F8'   ED 5B 0017"             ld      de,(lcptr)
07FC'   21 294D"                ld      hl,symtbl+(symmax-1)*symlen
07FF'   22 0017"                ld      (lcptr),hl
0802'   B7                      or      a
0803'   ED 52                   sbc     hl,de
0805'   44                      ld      b,h
0806'   4D                      ld      c,l
0807'   21 0010                 ld      hl,symlen
080A'   19                      add     hl,de
080B'   16 00                   ld      d,0
080D'   1E 01                   ld      e,1
080F'                   elisym1:
080F'   78                      ld      a,b
0810'   B1                      or      c
0811'   C8                      ret     z
0812'   1D                      dec     e
0813'   20 25                   jr      nz,elisym2
0815'   1E 10                   ld      e,symlen
0817'   CB 7E                   bit     7,(hl)          ;undefined ?
0819'   28 1F                   jr      z,elisym2
081B'   C5                      push    bc
081C'   D5                      push    de
081D'   E5                      push    hl
081E'   23                      inc     hl
081F'   E5                      push    hl
0820'   11 000C                 ld      de,idelen
0823'   19                      add     hl,de
0824'   36 00                   ld      (hl),0
0826'   21 083F'                ld      hl,udlms
0829'   CD 03F0'                call    prtmsg          ;print error message
082C'   E1                      pop     hl
082D'   CD 03F0'                call    prtmsg
0830'   2A 0000"                ld      hl,(errcou)
0833'   23                      inc     hl
0834'   22 0000"                ld      (errcou),hl
0837'   E1                      pop     hl
0838'   D1                      pop     de
0839'   C1                      pop     bc
083A'                   elisym2:
083A'   72                      ld      (hl),d
083B'   23                      inc     hl
083C'   0B                      dec     bc
083D'   18 D0                   jr      elisym1
                        ;
083F'   0D 0A 20 20     udlms:  defb    cr,lf,'    ?: Undefined label : ',0
0843'   20 20 3F 3A
0847'   20 55 6E 64
084B'   65 66 69 6E
084F'   65 64 20 6C
0853'   61 62 65 6C
0857'   20 3A 20 00
                        ;
                        ;       **  clear symbol table  **
                        ;
085B'                   clrstb::
085B'   21 095D"                ld      hl,symtbl
085E'   22 0015"                ld      (glptr),hl
0861'   01 2000                 ld      bc,symmax*symlen
0864'                   clrstbl:
0864'   36 00                   ld      (hl),0
0866'   23                      inc     hl
0867'   0B                      dec     bc
0868'   78                      ld      a,b
0869'   B1                      or      c
086A'   20 F8                   jr      nz,clrstbl
086C'   21 294D"                ld      hl,symtbl+(symmax-1)*symlen
086F'   22 0017"                ld      (lcptr),hl
```

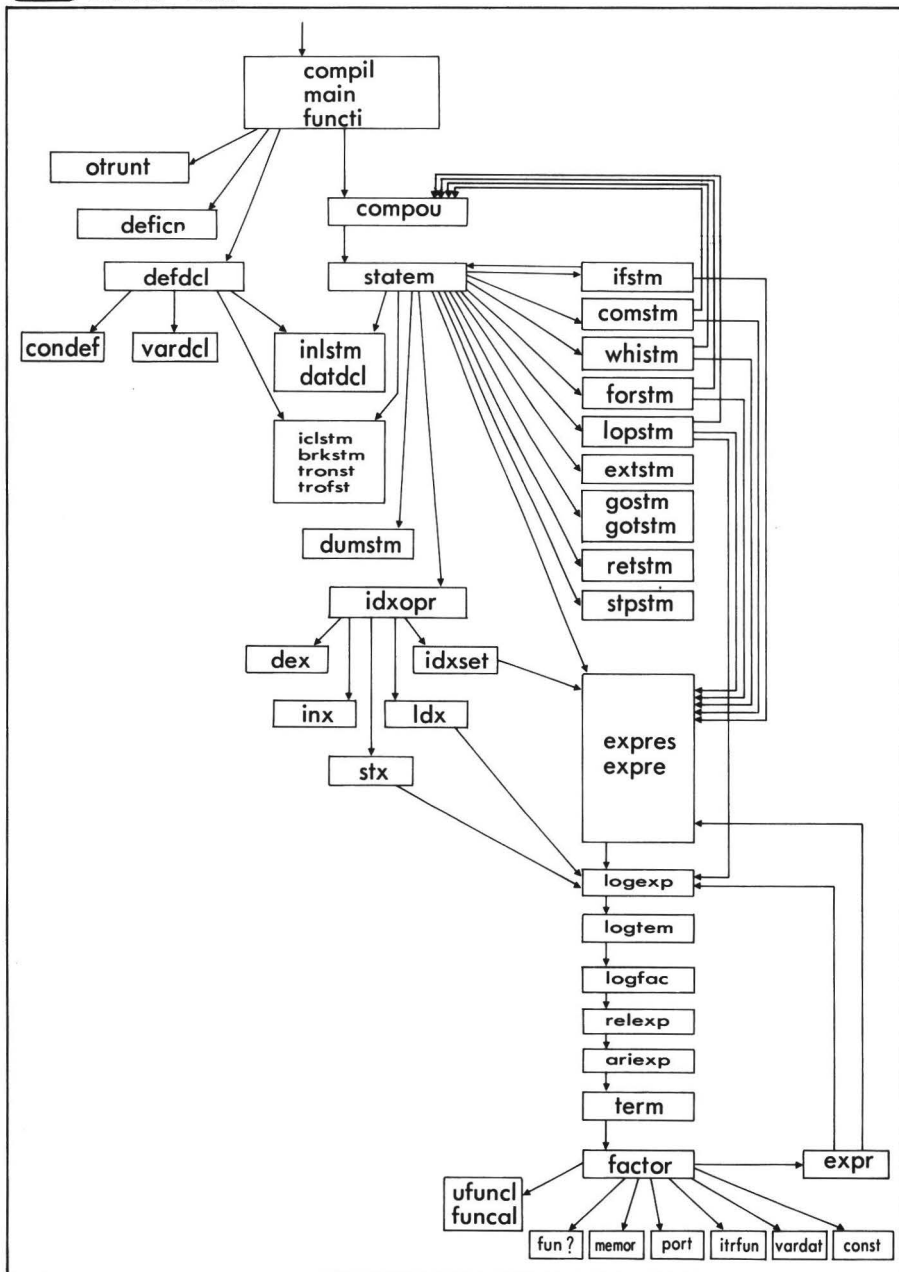
```
0872'   C9                          ret
                            ;
                            ;       **  put global symbol  **
                            ;
0873'                       putglo::
0873'   11 095D"                    ld      de,symtbl
0876'   2A 0015"                    ld      hl,(glptr)
0879'   B7                          or      a
087A'   ED 52                       sbc     hl,de
087C'   D5                          push    de
087D'   DD E1                       pop     ix
087F'                       putglo1:
087F'   7C                          ld      a,h
0880'   B5                          or      l
0881'   28 5E                       jr      z,putglo5
0883'   E5                          push    hl
0884'   DD 6E 0D                    ld      l,(ix+idelen+1)
0887'   DD 66 0E                    ld      h,(ix+idelen+2)  ; hl = address
088A'   DD 36 0D 00                 ld      (ix+idelen+1).0
088E'   DD CB 00 7E                 bit     7,(ix)          ;undefined ?
0892'   28 19                       jr      z,putglo2
0894'   DD E5                       push    ix
0896'   21 08E7'                    ld      hl,udfms
0899'   CD 03F0'                    call    prtmsg          ;print error message
089C'   E1                          pop     hl
089D'   E5                          push    hl
089E'   23                          inc     hl
089F'   CD 03F0'                    call    prtmsg
08A2'   2A 0000"                    ld      hl,(errcou)
08A5'   23                          inc     hl
08A6'   22 0000"                    ld      (errcou),hl
08A9'   DD E1                       pop     ix
08AB'   18 29                       jr      putglo4
08AD'                       putglo2:
08AD'   06 58                       ld      b,58h
08AF'   DD 7E 00                    ld      a,(ix)
08B2'   E6 0F                       and     0fh
08B4'   3D                          dec     a               ;constant ?
08B5'   28 08                       jr      z,putglo3
08B7'   04                          inc     b
08B8'   3D                          dec     a               ;variable ?
08B9'   28 04                       jr      z,putglo3
08BB'   04                          inc     b
08BC'   3D                          dec     a               ;data ?
08BD'   20 17                       jr      nz,putglo4
08BF'                       putglo3:
08BF'   C5                          push    bc
08C0'   DD E5                       push    ix
08C2'   CD 0000*                    call    ptloc           ;put location address
08C5'   E1                          pop     hl
08C6'   F1                          pop     af
08C7'   E5                          push    hl
08C8'   23                          inc     hl
08C9'   11 0000*                    ld      de,strbuf
08CC'   01 000D                     ld      bc,idelen+1
08CF'   ED B0                       ldir                    ;move
08D1'   CD 0000*                    call    ptname          ;put name
08D4'   DD E1                       pop     ix
08D6'                       putglo4:
08D6'   11 0010                     ld      de,symlen
08D9'   DD 19                       add     ix,de
08DB'   E1                          pop     hl
08DC'   B7                          or      a

08DD'   ED 52                       sbc     hl,de
08DF'   18 9E                       jr      putglo1
                            ;
08E1'                       putglo5:
08E1'   2A 0000*                    ld      hl,(cloc)
08E4'   C3 0000*                    jp      ptloc           ;put new code loc
```

```
                          ;
08E7'    0D 0A 20 20     udfms:  defb    cr,lf,'    ?: Undefined function name : ',0
08EB'    20 20 3F 3A
08EF'    20 55 6E 64
08F3'    65 66 69 6E
08F7'    65 64 20 66
08FB'    75 6E 63 74
08FF'    69 6F 6E 20
0903'    6E 61 6D 65
0907'    20 3A 20 00

                          ;---------------------------------------------------
                          ;-               w o r k   a r e a                 -
                          ;---------------------------------------------------
090B'                                    dseg
0000"                     work.b::

0000"                     errcou::defs   2                :error counter

0002"                     slino:  defs   2                ;source line number

0004"                     linol:  defs   2                ;save area
0006"                     cptrl:  defs   2
0008"                     slinol: defs   2

000A"                     inclf:: defs   1                ; 0=source, 0ffh=include

000B"                     sbrptr::defs   2                ;source input buffer read pointer
000D"                     sbrlen::defs   2                ;source input buffer read length
000F"                     ibrptr::defs   2                ;include input buffer read pointer
0011"                     ibrlen::defs   2                ;include input buffer read length

0013"                     obsptr::defs   2                ;object output buffer store pointer

0015"                     glptr:: defs   2                ;global symbol store pointer
0017"                     lcptr:: defs   2                ;local  symbol store pointer
0019"                     ofcb::  defs   33               ;object fcb
003A"                     ifcb::  defs   33               ;include fcb

005B"                     sbmsad::defs   2                ;sub error message address

0080                      linbufs equ    128
005D"                     linbuf::defs   linbufs          ;line buffer
00DD"                     ilibuf::defs   linbufs

0200                      sbfsiz  equ    reclen*4
015D"                     souibf::defs   sbfsiz           ;source input buffer
0200                      ibfsiz  equ    reclen*4
035D"                     incibf::defs   ibfsiz
0400                      obfsiz  equ    reclen*8
055D"                     objobf::defs   obfsiz           ;object output buffer
095D"                     symtbl::defs   symmax*symlen    ;symbol table

                                  end    start
```

(2)　STECMP. MACの構成とアセンブル・リスト

STECMP. MACは実際にコンパイルを行う部分で，１パス方式です。構文解析法に再帰的な方法を用いているため，ほぼ1-3で示した構文図の通りに構文が解析され，そのつどオブジェクト・コードを生成するようになっています。図2-6はSTECMPの構成です。図中の各ルーチンは次のような処理を行っています。

compil：プログラムの頭書きの処理と全域的な名前の定義と宣言。
main：主プログラムの処理。
functi：二度目以降の全域的な名前の定義と宣言。関数の頭書きとブロックの処理。
otrunt：ランタイム・ルーチンをリロケーションし，オブジェクト・ファイルへ出力する。
deficn：表意定数の定義。
defdcl：定義と宣言。
condef：定数名の定義。
vardcl：変数名の宣言とメモリの割りつけ。
inlstm：inline 文の処理。
datdcl：データ名の宣言とデータの格納。
statem：ラベルおよび文（ステートメント）の処理。
iclstm：コンパイル制御文% include の処理。
brkstm：コンパイル制御文% break の処理。
tronst：コンパイル制御文% tron の処理。
trofst：コンパイル制御文% troff の処理。
comstm：複合文あるいは until 文の処理。
dumstm：空文の処理。
compou：複合文の処理。
whistm：while 文の処理。
forstm：for 文の処理。
lopstm：loop 文の処理。
ifstm：if 文の処理。
extstm：exit 文の処理。

図2-6 STECMP の構成

gostm および **gotstm**：goto 文の処理。
retstm ：return 文の処理。
stpstm ：stop 文の処理。
idxopr ：インデックス操作文の処理。
idxset ：set 文の処理。
ldx ：ldx 文の処理。
stx ：stx 文の処理。
inx ：inx 文の処理。
dex ：dex 文の処理。
expres：文としての式の処理。
expre ：式の処理。
logexp：論理式の処理。
logtem：論理項の処理。
logfac ：論理因子の処理。
relexp ：関係式の処理。
ariexp ：算術式の処理。
term ：項の処理。
factor ：因子の処理。
expr ：カンマ式の処理。
fuc? ：条件演算関数の処理。
memor：メモリ配列の参照，代入に関する処理。
port ：ポート配列の参照，代入に関する処理。
itrfun ：組み込み関数の処理。
vardat：変数，データの参照，あるいは変数への代入に関する処理。
const ：式の中で使用される定数の処理。
ufuncl ：まだシンボル・テーブルに登録されていない関数名を呼び出そうとする場合の処理。
funcal ：すでにシンボル・テーブルに登録されている関数名を呼び出すときの処理。

STECMP.MAC には，この他に各ルーチンから共通に使われるサブルーチンが数多くあります。このうち，主な

ものを次に示します。

ptjr　：オブジェクト・コードとしてリラティブ・ジャンプ命令が使えるようなら，リラティブ・ジャンプ命令を生成する。

propt　：式は本書第一部で解説した簡単な最適化を行っている。このルーチンはその最適化のための準備をする。

stoptb：式の最適化のためのテーブル（スタック構造になっている）に演算数を入れる。

decopt：式の最適化のためのテーブルから演算数を一つ消す（つまり，POP して捨てる）。

oprexc：式の最適化のためのテーブルの，最新の演算数と次の演算数を交換する。

dyagen（codgen）：二項演算のオブジェクト・コードを生成する。式の最適化のためのテーブルからは二つの演算数がなくなる。

ptcod1　：１バイトのオブジェクト・コードを出力。

ptcod2　：２バイトのオブジェクト・コードを出力。

ptcodw：１ワードのオブジェクト・コードを出力。

ptcod3　：３バイト命令（１バイト＋１ワード）のオブジェクト・コードを出力。

ptname：名前をオブジェクト・ファイルへ出力。

ptloc　：現在のロケーション・カウンタの値をオブジェクト・ファイルへ出力。

ptcha　：チェイン・アドレスをオブジェクト・ファイルへ出力。

conexp：定数式の処理。

bytcon：バイト定数を求める。

wodcon：ワード定数を求める。

gtoken：字句解析。

●リスト2-5　STECMPのアセンブル・リスト

```
                        title   Stellar compiler  compile module
                        subttl  Rev 1.01  08/30/1984
                        name    ('cmpmod')
                .z80
                ;***************************************************
                ;*                                                 *
                ;*       Stellar compiler    compile module        *
                ;*              ( Rev 1.01 )                       *
                ;*                                                 *
                ;*     Copyrigth (c) 1984 H.Ohnuki / MIA           *
                ;*                                                 *
                ;***************************************************

                ;---------------------------------------------------
                ;-              external symbol                    -
                ;---------------------------------------------------

                ;
                ;  i/o module
                ;
                        external        getsou,putobj
                        external        stptyp,retadr
                        external        defsta,dsaind
                        external        regsym,seasym,elisym
                        external        clrstb
                        external        putglo,prtmsg
                        external        abort,error
                        external        ddspfn,ddspli
                        external        inclu,edincl.pbreak.tron,troff
                ;
                ;  run time module
                ;
                        external        @start,@stop
                        external        @mul.m,@mul.r
                        external        @div.m,@div.r,@rem.m,@rem.r
                        external        @shl.m,@shl.r,@shr.m.@shr.r
                        external        @setag,@prog
                        external        @rtwork,@stptp,@retad,@temp,@var
                        external        crelat,drelat

                ;---------------------------------------------------
                ;-              p r o g r a m                      -
                ;---------------------------------------------------

0000'                           cseg

                ;
                ; start of compile
                ;
0000'           compil::
0000'   CD 0000*        call    getsou          ;get source program ( first line )
0003'   D8              ret     c
0004'   22 0000"        ld      (cptr),hl
0007'   ED 53 0002"     ld      (lino),de
000B'   CD 1B0F'        call    gtoken          ;get first token
000E'   FE 80           cp      80h             ;'prog' ?
0010'   C2 200E'        jp      nz,synerr
0013'   CD 1B0F'        call    gtoken
0016'   3D              dec     a
0017'   C2 200E'        jp      nz,synerr
001A'   CD 198C'        call    movname
                ;
                ; option switch
                ;
001D'   AF              xor     a
001E'   32 0004"        ld      (optisw).a      ;option switch all off
0021'   CD 1B0F'        call    gtoken
```

```
0024'  FE 28               cp      '('
0026'  C2 200E'            jp      nz,synerr
0029'              compil1:
0029'  CD 1B0F'            call    gtoken
002C'  FE 29               cp      ')'
002E'  CA 00AD'            jp      z,compil7
0031'  FE 25               cp      '%'
0033'  C2 1EBE'            jp      nz,opserr
0036'  CD 1B0F'            call    gtoken
0039'  FE 84               cp      84h          ;'debug' ?
003B'  20 0A               jr      nz,compil2
003D'  21 0004"            ld      hl,optisw
0040'  CB E6               set     4,(hl)       ;%debug sw on
0042'  CD 1B0F'            call    gtoken
0045'  18 5A               jr      compil6
0047'              compil2:
0047'  3D                  dec     a
0048'  C2 1EBE'            jp      nz,opserr
004B'  2A 0017"            ld      hl,(identi)
004E'  7C                  ld      a,h
004F'  B7                  or      a
0050'  C2 1EBE'            jp      nz,opserr
0053'  7D                  ld      a,l
0054'  CD 1DB5'            call    traupc       ;translate to upper case
0057'  F5                  push    af
0058'  CD 1B0F'            call    gtoken
005B'  FE 3A               cp      ':'
005D'  C2 1EBE'            jp      nz,opserr
0060'  F1                  pop     af
0061'  FE 50               cp      'P'          ;%p ?
0063'  20 09               jr      nz,compil3
0065'  21 0004"            ld      hl,optisw
0068'  CB FE               set     7,(hl)       ;%p sw on
006A'  06 80               ld      b,80h
006C'  18 19               jr      compil5
006E'              compil3:
006E'  FE 44               cp      'D'          ;%d ?
0070'  20 09               jr      nz,compil4
0072'  21 0004"            ld      hl,optisw
0075'  CB F6               set     6,(hl)
0077'  06 40               ld      b,40h
0079'  18 0C               jr      compil5
007B'              compil4:
007B'  FE 53               cp      'S'          ;%s ?
007D'  C2 1EBE'            jp      nz,opserr
0080'  21 0004"            ld      hl,optisw
0083'  CB EE               set     5,(hl)
0085'  06 20               ld      b,20h
0087'              compil5:
0087'  C5                  push    bc
0088'  CD 1B0F'            call    gtoken
008B'  CD 1A01'            call    wodcon       ;word constant
008E'  F1                  pop     af
008F'  87                  add     a,a
0090'  30 03               jr      nc,$+5
0092'  22 0006"             ld     (codorg),hl  ;set code segment origin address
0095'  87                  add     a,a
0096'  30 03               jr      nc,$+5
0098'  22 0008"             ld     (datorg),hl  ;set data segment origin address
009B'  87                  add     a,a
009C'  30 03               jr      nc,$+5
009E'  22 000A"             ld     (stkbot),hl  ;set stack bottom address
00A1'              compil6:
00A1'  3A 0036"            ld      a,(tokcod)
00A4'  FE 2C               cp      ','
00A6'  28 81               jr      z,compil1
00A8'  FE 29               cp      ')'
00AA'  C2 1EBE'            jp      nz,opserr
00AD'              compil7:
```

```
00AD'  CD 1B0F'          call   gtoken
00B0'  FE 3B             cp     ';'
00B2'  C2 200E'          jp     nz,synerr
00B5'  3A 0004"          ld     a,(optisw)
00B8'  CB 67             bit    4,a            ;%debug sw on ?
00BA'  3E 55             ld     a,55h
00BC'  28 02             jr     z,$+4
00BE'  3E 65               ld   a,65h
00C0'  CD 0000*          call   putobj         ;put start mark
00C3'  2A 0006"          ld     hl,(codorg)
00C6'  22 000C"          ld     (cloc),hl      ;set code location counter
00C9'  CD 19B2'          call   ptloc
00CC'  2A 0008"          ld     hl,(datorg)
00CF'  22 000E"          ld     (dloc),hl      ;set data location counter
00D2'  22 0010"          ld     (wloc),hl
00D5'  3E 56             ld     a,56h
00D7'  CD 199F'          call   ptname         ;put program name
00DA'  21 03F1'          ld     hl,pgnati
00DD'  CD 0000*          call   prtmsg
00E0'  21 0026"          ld     hl,strbuf
00E3'  CD 0000*          call   prtmsg         ;print program name
00E6'  CD 0417'          call   otrunt         ;output run time routine
00E9'  CD 0000*          call   clrstb         ;clear symbol table
                  ;
                  ; define figurative constant
                  ;
00EC'  2A 0008"          ld     hl,(datorg)
00EF'  CD 0487'          call   deficn
00F2'  5F 77 6F 72       defb   '_work',0
00F6'  6B 00
00F8'  2A 000E"          ld     hl,(dloc)
00FB'  CD 0487'          call   deficn
00FE'  5F 76 61 72       defb   '_var',0
0102'  00
0103'  2A 0006"          ld     hl,(codorg)
0106'  CD 0487'          call   deficn
0109'  5F 63 6F 64       defb   '_code',0
010D'  65 00
                  ;
                  ; main
                  ;
010F'             main::
010F'  16 3E             ld     d,3eh          ;" LD  A,STOP_TYPE  "
0111'  1E 00*            ld     e,stptyp
0113'  CD 1964'          call   ptcod2
0116'  3E 32             ld     a,32h          ;" LD  (@STPTY),A  "
0118'  2A 0008"          ld     hl,(datorg)
011B'  11 0000*          ld     de,@stptp-@rtwork
011E'  19                add    hl,de
011F'  CD 196E'          call   ptcod3
0122'  3E 00*            ld     a,stptyp
0124'  B7                or     a              ;halt ?
0125'  28 22             jr     z,main0
0127'  3D                dec    a              ;ret ?
0128'  20 08             jr     nz,main0.5
012A'  11 ED73           ld     de,0ed73h      ;" LD  (@RETAD),SP  "
012D'  CD 1964'          call   ptcod2
0130'  18 0D             jr     main0.6
0132'             main0.5:
0132'  3E 21             ld     a,021h         ;" LD  HL,RETURN_ADDRESS  "
0134'  21 0000*          ld     hl,retadr
0137'  CD 196E'          call   ptcod3
013A'  3E 22             ld     a,22h          ;" LD  (@RETAD),HL  "
013C'  CD 1953'          call   ptcod1
013F'             main0.6:
013F'  2A 0008"          ld     hl,(datorg)
0142'  11 0000*          ld     de,@retad-@rtwork
0145'  19                add    hl,de
0146'  CD 1966'          call   ptcodw
```

```
0149'                   main0:
0149'   3A 0004"                ld      a,(optisw)
014C'   2A 000A"                ld      hl,(stkbot)
014F'   CB 6F                   bit     5,a             ;%s on ?
0151'   20 15                   jr      nz,main1
0153'   3E 00*                  ld      a,dsaind
0155'   21 0000*                ld      hl,defsta
0158'   B7                      or      a               ;indirect ?
0159'   28 0D                   jr      z,main1
015B'   E5                      push    hl
015C'   11 ED7B                 ld      de,0ed7bh       ;"  LD  SP,(DEFSTA) "
015F'   CD 1964'                call    ptcod2
0162'   E1                      pop     hl
0163'   CD 1966'                call    ptcodw
0166'   18 05                   jr      main2
0168'                   main1:
0168'   3E 31                   ld      a,31h           ;"  LD  SP,nn  "
016A'   CD 196E'                call    ptcod3
016D'                   main2:
016D'   3E 21                   ld      a,21h           ;"  LD  HL,@STOP "
016F'   2A 0006"                ld      hl,(codorg)
0172'   11 0000*                ld      de,@stop-@start
0175'   19                      add     hl,de
0176'   CD 196E'                call    ptcod3
0179'   3E E5                   ld      a,0e5h          ;"  PUSH HL  "
017B'   CD 1953'                call    ptcod1
                        ;
017E'   CD 1B0F'                call    gtoken
0181'   AF                      xor     a
0182'   32 008C"                ld      (idxpsf),a
0185'   CD 049E'                call    defdcl          ;global constant,variable,data
0188'   FE 7B                   cp      '{'
018A'   C2 200E'                jp      nz,synerr
018D'   3E FF                   ld      a,0ffh
018F'   32 0005"                ld      (stmflg),a
0192'   2A 0002"                ld      hl,(lino)
0195'   CD 0000*                call    ddspli          ;line number display object out
0198'   CD 0B45'                call    pshlea
019B'   CD 0788'                call    compou          ;{ statement ... }
019E'   CD 0B5D'                call    poplea
01A1'   AF                      xor     a
01A2'   32 0005"                ld      (stmflg),a
01A5'   CD 1B0F'                call    gtoken
01A8'   3E C9                   ld      a,0c9h          ;"  RET  "
01AA'   CD 1953'                call    ptcod1
                        ;
                        ; function
                        ;
01AD'                   functi::
01AD'   CD 0000*                call    elisym          ;elimination local symbol
01B0'   3A 0036"                ld      a,(tokcod)
01B3'   3C                      inc     a               ;end ?
01B4'   CA 03E8'                jp      z,endcom
01B7'   AF                      xor     a
01B8'   32 008C"                ld      (idxpsf),a
01BB'   67                      ld      h,a
01BC'   6F                      ld      l,a
01BD'   22 008D"                ld      (retjad),hl
01C0'   F5                      push    af
01C1'   CD 049E'                call    defdcl          ;global constant,variable,data
01C4'   2F                      cpl
01C5'   32 0037"                ld      (localf),a
01C8'   3A 0036"                ld      a,(tokcod)
01CB'   FE FF                   cp      0ffh            ;end ?
01CD'   20 04                   jr      nz,functi0
01CF'   F1                      pop     af
01D0'   C3 03E8'                jp      endcom
01D3'                   functi0:
01D3'   FE 8F                   cp      8fh             ;'recursive' ?
```

```
01D5'   20 07               jr      nz,functi1
01D7'   F1                  pop     af
01D8'   F6 80               or      80h
01DA'   F5                  push    af
01DB'   CD 1B0F'            call    gtoken
01DE'               functi1:
01DE'   3D                  dec     a               ;identifier ?
01DF'   C2 1ED7'            jp      nz,ilferr
01E2'   21 0016"            ld      hl,idetyp
01E5'   CD 0000*            call    seasym          ;search symbol table
01E8'   06 09               ld      b,09h
01EA'   B7                  or      a               ;found ?
01EB'   20 16               jr      nz,functi2
01ED'   3A 0016"            ld      a,(idetyp)
01F0'   FE 89               cp      89h             ;function ?
01F2'   28 07               jr      z,functi1.5
01F4'   C5                  push    bc
01F5'   CD 1EF4'            call    ilefun
01F8'   C1                  pop     bc
01F9'   18 08               jr      functi2
01FB'               functi1.5:
01FB'   2A 0023"            ld      hl,(adrs)
01FE'   F5                  push    af
01FF'   CD 19B7'            call    ptcha           ;put chain address
0202'   C1                  pop     bc
0203'               functi2:
0203'   2A 000C"            ld      hl,(cloc)
0206'   22 0023"            ld      (adrs),hl
0209'   CB B8               res     7,b
020B'   21 0016"            ld      hl,idetyp
020E'   70                  ld      (hl),b
020F'   CD 0000*            call    regsym          ;symbol table register
0212'   CD 198C'            call    movname
0215'   3E 57               ld      a,57h
0217'   CD 199F'            call    ptname          ;put function name
021A'   21 0404'            ld      hl,funati
021D'   CD 0000*            call    prtmsg
0220'   21 0026"            ld      hl,strbuf
0223'   E5                  push    hl
0224'   CD 0000*            call    prtmsg          ;print function name
0227'   E1                  pop     hl
0228'   CD 0000*            call    ddspfn          ;function name display object out
022B'   CD 1B0F'            call    gtoken
022E'   FE 28               cp      '('
0230'   C2 200E'            jp      nz,synerr
0233'   2A 000E"            ld      hl,(dloc)
0236'   22 0088"            ld      (varadr),hl
0239'   AF                  xor     a
023A'   32 008A"            ld      (varsiz),a
023D'   CD 1B0F'            call    gtoken
0240'   FE 3B               cp      ';'
0242'   28 44               jr      z,functi5
0244'   FE 29               cp      ')'
0246'   28 7B               jr      z,functi10
0248'   3D                  dec     a               ;identifier ?
0249'   C2 1F06'            jp      nz,ilnerr
024C'   18 03               jr      $+5
024E'               functi3:
024E'   CD 1B0F'              call  gtoken
0251'   CD 0662'            call    symchk          ;symbol search & type check
0254'   F6 02               or      02h
0256'   2A 000E"            ld      hl,(dloc)
0259'   22 0023"            ld      (adrs).hl
025C'   21 0016"            ld      hl.idetyp
025F'   77                  ld      (hl).a
0260'   CD 0000*            call    regsym          ;symbol table register
0263'   2A 000E"            ld      hl.(dloc)
0266'   23                  inc     hl
0267'   22 000E"            ld      (dloc).hl
```

```
026A'   22 0010"                ld      (wloc),hl
026D'   21 008A"                ld      hl,varsiz
0270'   7E                      ld      a,(hl)
0271'   FE 20                   cp      32
0273'   30 01                   jr      nc,$+3
0275'   34                        inc   (hl)
0276'   CD 1B0F'                call    gtoken
0279'   FE 2C                   cp      ','
027B'   28 D1                   jr      z,functi3
027D'   FE 3B                   cp      ';'
027F'   28 07                   jr      z,functi5
0281'   FE 29                   cp      ')'
0283'   28 3E                   jr      z,functi10
0285'   C3 200E'                jp      synerr
0288'                   functi5:
0288'   CD 1B0F'                call    gtoken
028B'   FE 23                   cp      '#'
028D'   20 07                   jr      nz,functi6
028F'   F1                      pop     af
0290'   F6 08                   or      1000b
0292'   F5                      push    af
0293'   CD 1B0F'                call    gtoken
0296'                   functi6:
0296'   FE BF                   cp      0bfh            ;'ix' ?
0298'   20 07                   jr      nz,functi7
029A'   F1                      pop     af
029B'   F6 04                   or      0100b
029D'   F5                      push    af
029E'   CD 1B0F'                call    gtoken
02A1'                   functi7:
02A1'   FE 2C                   cp      ','
02A3'   20 19                   jr      nz,functi9
02A5'   CD 1B0F'                call    gtoken
02A8'   FE 23                   cp      '#'
02AA'   20 07                   jr      nz,functi8
02AC'   F1                      pop     af
02AD'   F6 02                   or      0010b
02AF'   F5                      push    af
02B0'   CD 1B0F'                call    gtoken
02B3'                   functi8:
02B3'   FE C0                   cp      0c0h            ;'iy' ?
02B5'   20 07                   jr      nz,functi9
02B7'   F1                      pop     af
02B8'   F6 01                   or      0001b
02BA'   F5                      push    af
02BB'   CD 1B0F'                call    gtoken
02BE'                   functi9:
02BE'   FE 29                   cp      ')'
02C0'   C2 200E'                jp      nz,synerr
02C3'                   functi10:
02C3'   F1                      pop     af
02C4'   CB 57                   bit     2,a
02C6'   28 0E                   jr      z,functi11
02C8'   F5                      push    af
02C9'   3E 22                   ld      a,22h           ;"  LD   (TEMP1),HL  "
02CB'   2A 0008"                ld      hl,(datorg)
02CE'   11 0000*                ld      de,@temp-@rtwork
02D1'   19                      add     hl,de
02D2'   CD 196E'                call    ptcod3
02D5'   F1                      pop     af
02D6'                   functi11:
02D6'   CB 47                   bit     0,a
02D8'   28 12                   jr      z,functi12
02DA'   F5                      push    af
02DB'   11 ED53                 ld      de,0ed53h       ;"  LD   (TEMP2),DE  "
02DE'   CD 1964'                call    ptcod2
02E1'   2A 0008"                ld      hl,(datorg)
02E4'   11 0002*                ld      de,@temp+2-@rtwork
02E7'   19                      add     hl,de
```

```
02E8'   CD 1966'            call    ptcodw
02EB'   F1                  pop     af
02EC'               functil2:
02EC'   F5                  push    af
02ED'   3A 008A"            ld      a,(varsiz)
02F0'   5F                  ld      e,a
02F1'   16 0E               ld      d,0eh           ;"  LD   C.n  "
02F3'   CD 1964'            call    ptcod2
02F6'   3E 11               ld      a,11h           ;"  LD   DE,0  "
02F8'   21 0000             ld      hl,0
02FB'   CD 196E'            call    ptcod3
02FE'   2A 0088"            ld      hl,(varadr)
0301'   F1                  pop     af
0302'   CB 7F               bit     7,a
0304'   28 08               jr      z,functil3
0306'   E5                  push    hl
0307'   ED 5B 000C"         ld      de,(cloc)
030B'   1B                  dec     de
030C'   1B                  dec     de
030D'   D5                  push    de
030E'               functil3:
030E'   F5                  push    af
030F'   3E 21               ld      a,21h           ;"  LD   HL,nn  "
0311'   CD 196E'            call    ptcod3
0314'   3E CD               ld      a,0cdh          ;"  CALL @SETAG  "
0316'   2A 0006"            ld      hl,(codorg)
0319'   11 0000*            ld      de,@setag-@start
031C'   19                  add     hl,de
031D'   CD 196E'            call    ptcod3
0320'   F1                  pop     af
0321'   CB 5F               bit     3,a
0323'   20 0D               jr      nz,functil4
0325'   F5                  push    af
0326'   11 DDE5             ld      de,0dde5h       ;"  PUSH IX  "
0329'   CD 1964'            call    ptcod2
032C'   3E FF               ld      a,0ffh
032E'   32 008C"            ld      (idxpsf),a
0331'   F1                  pop     af
0332'               functil4:
0332'   CB 4F               bit     1,a
0334'   20 0D               jr      nz,functil5
0336'   F5                  push    af
0337'   11 FDE5             ld      de,0fde5h       ;"  PUSH IY  "
033A'   CD 1964'            call    ptcod2
033D'   3E FF               ld      a,0ffh
033F'   32 008C"            ld      (idxpsf),a
0342'   F1                  pop     af
0343'               functil5:
0343'   CB 57               bit     2,a
0345'   28 12               jr      z,functil6
0347'   F5                  push    af
0348'   11 DD2A             ld      de,0dd2ah       ;"  LD   IX,(TEMP1) "
034B'   CD 1964'            call    ptcod2
034E'   2A 0008"            ld      hl,(datorg)
0351'   11 0000*            ld      de,@temp-@rtwork
0354'   19                  add     hl,de
0355'   CD 1966'            call    ptcodw
0358'   F1                  pop     af
0359'               functil6:
0359'   CB 47               bit     0,a
035B'   28 12               jr      z,functil7
035D'   F5                  push    af
035E'   11 FD2A             ld      de,0fd2ah       ;"  LD   IY,(TEMP2) "
0361'   CD 1964'            call    ptcod2
0364'   2A 0008"            ld      hl,(datorg)
0367'   11 0002*            ld      de,@temp+2-@rtwork
036A'   19                  add     hl,de
036B'   CD 1966'            call    ptcodw
036E'   F1                  pop     af
```

```
036F'                   functi17:
036F'   F5                      push    af
0370'   CD 1B0F'                call    gtoken
0373'   FE 3B                   cp      ';'
0375'   C2 200E'                jp      nz,synerr
                        ;
0378'   CD 1B0F'                call    gtoken
037B'   3E FF                   ld      a,0ffh
037D'   CD 049E'                call    defdcl          ;local constant,variable,data
0380'   FE 7B                   cp      '{'
0382'   C2 200E'                jp      nz,synerr
0385'   3E FF                   ld      a,0ffh
0387'   32 0005"                ld      (stmflg),a
038A'   2A 0002"                ld      hl,(lino)
038D'   CD 0000*                call    ddspli          ;line number display object out
0390'   CD 0B45'                call    pshlea
0393'   CD 0788'                call    compou          ;{ statement ... }
0396'   CD 0B5D'                call    poplea
0399'   AF                      xor     a
039A'   32 0005"                ld      (stmflg),a
039D'   CD 1B0F'                call    gtoken
                        ;
03A0'   2A 008D"                ld      hl,(retjad)
03A3'   7C                      ld      a,h
03A4'   B5                      or      l
03A5'   C4 19B7'                call    nz,ptcha        ;put chain address
03A8'   F1                      pop     af
03A9'   CB 4F                   bit     1,a
03AB'   20 08                   jr      nz,functi18
03AD'   F5                      push    af
03AE'   11 FDE1                 ld      de,0fde1h       ;"  POP  IY  "
03B1'   CD 1964'                call    ptcod2
03B4'   F1                      pop     af
03B5'                   functi18:
03B5'   CB 5F                   bit     3,a
03B7'   20 08                   jr      nz,functi19
03B9'   F5                      push    af
03BA'   11 DDE1                 ld      de,0dde1h       ;"  POP  IX  "
03BD'   CD 1964'                call    ptcod2
03C0'   F1                      pop     af
03C1'                   functi19:
03C1'   F5                      push    af
03C2'   3E C9                   ld      a,0c9h          ;"  RET  "
03C4'   CD 1953'                call    ptcod1
03C7'   F1                      pop     af
03C8'   87                      add     a,a
03C9'   D2 01AD'                jp      nc,functi
03CC'   E1                      pop     hl
03CD'   CD 19B2'                call    ptloc           ;put location count
03D0'   D1                      pop     de
03D1'   2A 000E"                ld      hl,(dloc)
03D4'   B7                      or      a
03D5'   ED 52                   sbc     hl,de
03D7'   CD 1966'                call    ptcodw          ;put word code
03DA'   2A 000C"                ld      hl,(cloc)
03DD'   2B                      dec     hl
03DE'   2B                      dec     hl
03DF'   22 000C"                ld      (cloc),hl
03E2'   CD 19B2'                call    ptloc           ;put location count
03E5'   C3 01AD'                jp      functi
                        ;
                        ; end of compile
                        ;
03E8'                   endcom::
03E8'   CD 0000*                call    putglo          ;put global symbol
03EB'   3E FF                   ld      a,0ffh          ;put end mark
03ED'   CD 0000*                call    putobj
03F0'   C9                      ret
                        ;
```

```
03F1'   0D 0A 50 72   pgnati: defb    13,10,'Program  name : ',0
03F5'   6F 67 72 61
03F9'   6D 20 20 6E
03FD'   61 6D 65 20
0401'   3A 20 00
0404'   0D 0A 46 75   funati: defb    13,10,'Function name : ',0
0408'   6E 63 74 69
040C'   6F 6E 20 6E
0410'   61 6D 65 20
0414'   3A 20 00
                      ;
                      ;       **  run time routine relocating out  **
                      ;
0417'                 otrunt::
0417'   01 0000*              ld      bc,@prog-@start
041A'   21 0000*              ld      hl,@start
041D'   DD 21 0000*           ld      ix,crelat
0421'   FD 21 0000*           ld      iy,drelat
0425'                 otrunt1:
0425'   7D                    ld      a,l
0426'   DD BE 00              cp      (ix)            ;relocate ?
0429'   20 1B                 jr      nz,otrunt2
042B'   7C                    ld      a,h
042C'   DD BE 01              cp      (ix+1)
042F'   20 15                 jr      nz,otrunt2
0431'   DD 23                 inc     ix
0433'   DD 23                 inc     ix
0435'   11 0000*              ld      de,@start       ;relocating code segment address
0438'   7E                    ld      a,(hl)
0439'   93                    sub     e
043A'   5F                    ld      e,a
043B'   23                    inc     hl
043C'   7E                    ld      a,(hl)
043D'   9A                    sbc     a,d
043E'   57                    ld      d,a
043F'   E5                    push    hl
0440'   2A 0006"              ld      hl,(codorg)
0443'   19                    add     hl,de
0444'   18 29                 jr      otrunt4
0446'                 otrunt2:
0446'   7D                    ld      a,l
0447'   FD BE 00              cp      (iy)            ;relocate ?
044A'   20 1B                 jr      nz,otrunt3
044C'   7C                    ld      a,h
044D'   FD BE 01              cp      (iy+1)
0450'   20 15                 jr      nz,otrunt3
0452'   FD 23                 inc     iy
0454'   FD 23                 inc     iy
0456'   11 0000*              ld      de,@rtwork      ;relocating data segment address
0459'   7E                    ld      a,(hl)
045A'   93                    sub     e
045B'   5F                    ld      e,a
045C'   23                    inc     hl
045D'   7E                    ld      a,(hl)
045E'   9A                    sbc     a,d
045F'   57                    ld      d,a
0460'   E5                    push    hl
0461'   2A 0008"              ld      hl,(datorg)
0464'   19                    add     hl,de
0465'   18 08                 jr      otrunt4
0467'                 otrunt3:
0467'   7E                    ld      a,(hl)
0468'   E5                    push    hl
0469'   C5                    push    bc
046A'   CD 1953'              call    ptcod1          ;put byte code
046D'   18 05                 jr      otrunt5
046F'                 otrunt4:
046F'   0B                      dec   bc
0470'   C5                      push  bc
```

```
0471'   CD 1966'              call   ptcodw          ;put word code
0474'                  otrunt5:
0474'   C1                    pop    bc
0475'   E1                    pop    hl
0476'   23                    inc    hl
0477'   0B                    dec    bc
0478'   78                    ld     a,b
0479'   B1                    or     c
047A'   20 A9                 jr     nz,otrunt1
                       ;
047C'   2A 000E"              ld     hl,(dloc)       ;reserve run time work
047F'   11 0000*              ld     de,@var-@rtwork
0482'   19                    add    hl,de
0483'   22 000E"              ld     (dloc),hl
0486'   C9                    ret
                       ;
                       ;      **  define figurative constant  **
                       ;
0487'                  deficn:: ;      hl          : constant value
                                ;      m(sp+1,sp): constant name address
0487'   22 0023"              ld     (adrs),hl
048A'   21 0016"              ld     hl,idetyp
048D'   36 01                 ld     (hl),01h        ;global constant
048F'   D1                    pop    de
0490'                  deficn1:
0490'   1A                    ld     a,(de)
0491'   13                    inc    de
0492'   23                    inc    hl
0493'   77                    ld     (hl),a
0494'   B7                    or     a
0495'   20 F9                 jr     nz,deficn1
0497'   D5                    push   de
0498'   21 0016"              ld     hl,idetyp
049B'   C3 0000*              jp     regsym          ;symbol table register
                       ;
                       ;      **  constant definition & variable, data declare  **
                       ;
049E'                  defdcl:: ;      a : 00h=blobal, not 00h=local
049E'   32 0037"              ld     (localf),a
04A1'                  defdcl1:
04A1'   3A 0036"              ld     a,(tokcod)
04A4'   FE 25                 cp     '%'             ;%include,%break,%tron,%troff ?
04A6'   20 08                 jr     nz,defdcl2
04A8'   CD 1B0F'              call   gtoken
04AB'   CD 0715'              call   iclstm
04AE'   18 F1                 jr     defdcl1
04B0'                  defdcl2:
04B0'   FE 8C                 cp     8ch             ;'cons' ?
04B2'   20 05                 jr     nz,defdcl3
04B4'   CD 04CA'              call   condef
04B7'   18 E8                 jr     defdcl1
04B9'                  defdcl3:
04B9'   FE 8D                 cp     8dh             ;'var' ?
04BB'   20 05                 jr     nz,defdcl4
04BD'   CD 050F'              call   vardcl
04C0'   18 DF                 jr     defdcl1
04C2'                  defdcl4:
04C2'   FE 8E                 cp     8eh             ;'data' ?
04C4'   C0                    ret    nz
04C5'   CD 05AC'              call   datdcl
04C8'   18 D7                 jr     defdcl1
                       ;
                       ; constant definition
                       ;
04CA'                  condef::
04CA'   CD 1B0F'              call   gtoken
04CD'   3D                    dec    a               ;identifier ?
04CE'   C2 1F06'              jp     nz,ilnerr
04D1'   CD 0662'              call   symchk          ;symbol search & type check
```

```
04D4'   F6 01                   or      01h             ;type = constant name
04D6'   01 000D                 ld      bc.adrs-idetyp
04D9'   21 0016"                ld      hl.idetyp
04DC'   77                      ld      (hl).a
04DD'   CD 1AF3'                call    blkpsh          ;block push
04E0'   CD 1B0F'                call    gtoken
04E3'   FE F0                   cp      0f0h            ;':=' ?
04E5'   C2 200E'                jp      nz.synerr
04E8'   CD 1B0F'                call    gtoken
04EB'   CD 19C7'                call    conexp          ;constant expression
04EE'   22 0023"                ld      (adrs).hl
04F1'   01 000D                 ld      bc.adrs-idetyp
04F4'   21 0016"                ld      hl.idetyp
04F7'   CD 1B03'                call    blkpop          ;block pop
04FA'   21 0016"                ld      hl.idetyp
04FD'   CD 0000*                call    regsym          ;symbol table register
0500'   3A 0036"                ld      a.(tokcod)
0503'   FE 2C                   cp      ','
0505'   28 C3                   jr      z.condef
0507'   FE 3B                   cp      ';'
0509'   C2 200E'                jp      nz.synerr
050C'   C3 1B0F'                jp      gtoken
                        ;
                        ; variable declare
                        ;
050F'                   vardcl::
050F'   CD 1B0F'                call    gtoken
0512'   3D                      dec     a               ;identifier ?
0513'   C2 1F06'                jp      nz.ilnerr
0516'   CD 0662'                call    symchk          ;symbol search & type check
0519'   F6 02                   or      02h             ;type = variable name
051B'   2A 000E"                ld      hl,(dloc)
051E'   22 0088"                ld      (varadr),hl
0521'   21 0001                 ld      hl,1
0524'   22 008A"                ld      (varsiz),hl
0527'   01 000D                 ld      bc,adrs-idetyp
052A'   21 0016"                ld      hl,idetyp
052D'   77                      ld      (hl),a
052E'   CD 1AF3'                call    blkpsh          ;block push
0531'   CD 1B0F'                call    gtoken
0534'   FE 5B                   cp      '['
0536'   20 14                   jr      nz,vardcl1
0538'   CD 1B0F'                call    gtoken
053B'   CD 19C7'                call    conexp          ;constant expression
053E'   22 008A"                ld      (varsiz),hl
0541'   3A 0036"                ld      a,(tokcod)
0544'   FE 5D                   cp      ']'
0546'   C2 200E'                jp      nz,synerr
0549'   CD 1B0F'                call    gtoken
054C'                   vardcl1:
054C'   FE BE                   cp      0beh            ;'at' ?
054E'   20 22                   jr      nz,vardcl2
0550'   CD 1B0F'                call    gtoken
0553'   FE 28                   cp      '('
0555'   C2 200E'                jp      nz,synerr
0558'   CD 1B0F'                call    gtoken
055B'   CD 19C7'                call    conexp          ;constant expression
055E'   22 0088"                ld      (varadr),hl
0561'   21 0000                 ld      hl.0
0564'   22 008A"                ld      (varsiz),hl
0567'   3A 0036"                ld      a,(tokcod)
056A'   FE 29                   cp      ')'
056C'   C2 200E'                jp      nz,synerr
056F'   CD 1B0F'                call    gtoken
0572'                   vardcl2:
0572'   2A 0088"                ld      hl,(varadr)
0575'   22 0023"                ld      (adrs).hl
0578'   01 000D                 ld      bc,adrs-idetyp
057B'   21 0016"                ld      hl,idetyp
```

```
057E'   CD 1B03'              call   blkpop          ;block pop
0581'   21 0016"              ld     hl,idetyp
0584'   CD 0000*              call   regsym          ;symbol table register
0587'   2A 000E"              ld     hl,(dloc)
058A'   ED 5B 008A"           ld     de,(varsiz)
058E'   19                    add    hl,de
058F'   22 000E"              ld     (dloc),hl
0592'   22 0010"              ld     (wloc),hl
0595'   3A 0036"              ld     a,(tokcod)
0598'   FE 2C                 cp     ','
059A'   CA 050F'              jp     z,vardcl
059D'   FE 3B                 cp     ';'
059F'   C2 200E'              jp     nz,synerr
05A2'   C3 1B0F'              jp     gtoken
                      ;
                      ; data declare & inline statement
                      ;
05A5'                 inlstm::
05A5'   3E 02                 ld     a,2
05A7'   32 0038"              ld     (datinl),a
05AA'   18 18                 jr     datdcl2
                      ;
05AC'                 datdcl::
05AC'   3E 01                 ld     a,1
05AE'   32 0038"              ld     (datinl),a
05B1'   3E C3                 ld     a,0c3h          ;"  JP   0  "
05B3'   21 0000               ld     hl,0
05B6'   CD 196E'              call   ptcod3
05B9'   2A 000C"              ld     hl,(cloc)
05BC'   2B                    dec    hl
05BD'   2B                    dec    hl
05BE'   22 0084"              ld     (skpadr),hl
05C1'                 datdcl1:
05C1'   CD 1B0F'              call   gtoken
05C4'                 datdcl2:
05C4'   3A 0038"              ld     a,(datinl)
05C7'   3D                    dec    a               ;data declare ?
05C8'   20 27                 jr     nz,datdcl3
05CA'   3A 0036"              ld     a,(tokcod)
05CD'   3D                    dec    a
05CE'   20 21                 jr     nz,datdcl3
05D0'   2A 0000"              ld     hl,(cptr)
05D3'   7E                    ld     a,(hl)
05D4'   FE 3A                 cp     ':'             ;data name ?
05D6'   20 19                 jr     nz,datdcl3
05D8'   23                    inc    hl
05D9'   22 0000"              ld     (cptr),hl
05DC'   CD 0662'              call   symchk          ;symbol search & type check
05DF'   F6 03                 or     03h             ;type = data name
05E1'   2A 000C"              ld     hl,(cloc)
05E4'   22 0023"              ld     (adrs),hl
05E7'   21 0016"              ld     hl,idetyp
05EA'   77                    ld     (hl),a
05EB'   CD 0000*              call   regsym          ;symbol table register
05EE'   CD 1B0F'              call   gtoken
05F1'                 datdcl3:
05F1'   3A 0036"              ld     a,(tokcod)
05F4'   FE 22                 cp     '"'             ;string ?
05F6'   20 2E                 jr     nz,datdcl6
05F8'   2A 0000"              ld     hl,(cptr)
05FB'                 datdcl4:
05FB'   7E                    ld     a,(hl)
05FC'   23                    inc    hl
05FD'   FE 22                 cp     '"'
05FF'   20 0F                 jr     nz,datdcl5
0601'   7E                    ld     a,(hl)
0602'   23                    inc    hl
0603'   FE 22                 cp     '"'
0605'   28 09                 jr     z,datdcl5
```

```
0607'   2B                      dec     hl
0608'   22 0000"                ld      (cptr),hl
060B'   CD 1B0F'                call    gtoken
060E'   18 35                   jr      datdcl8
0610'                   datdcl5:
0610'   B7                      or      a
0611'   20 0C                   jr      nz,datdcl5.5
0613'   2B                      dec     hl
0614'   22 0000"                ld      (cptr),hl
0617'   CD 1F48'                call    bstrdt
061A'   CD 1B0F'                call    gtoken
061D'   18 26                   jr      datdcl8
061F'                   datdcl5.5:
061F'   E5                      push    hl
0620'   CD 1953'                call    ptcodl          ;put byte object
0623'   E1                      pop     hl
0624'   18 D5                   jr      datdcl4
0626'                   datdcl6:
0626'   FE 23                   cp      '#'
0628'   28 04                   jr      z,$+6
062A'   FE 92                     cp    92h             ;'word' ?
062C'   20 0B                     jr    nz,datdcl7
062E'   CD 1B0F'                call    gtoken
0631'   CD 1A01'                call    wodcon          ;word constant
0634'   CD 1966'                call    ptcodw          ;put word object
0637'   18 0C                   jr      datdcl8
0639'                   datdcl7:
0639'   FE 91                   cp      91h             ;'byte' ?
063B'   CC 1B0F'                call    z,gtoken
063E'   CD 19ED'                call    bytcon          ;byte constant
0641'   7D                      ld      a,l
0642'   CD 1953'                call    ptcodl          ;put byte object
0645'                   datdcl8:
0645'   3A 0036"                ld      a,(tokcod)
0648'   FE 2C                   cp      ','
064A'   CA 05C1'                jp      z,datdcl1
064D'   FE 3B                   cp      ';'
064F'   C2 200E'                jp      nz,synerr
0652'   21 0038"                ld      hl,datinl
0655'   7E                      ld      a,(hl)
0656'   36 00                   ld      (hl),0
0658'   3D                      dec     a               ;data declare ?
0659'   2A 0084"                ld      hl,(skpadr)
065C'   CC 19B7'                call    z,ptcha         ;put chain address
065F'   C3 1B0F'                jp      gtoken
                        ;
                        ;       search symbol table & type check
                        ;
0662'                   symchk::
0662'   21 0016"                ld      hl,idetyp
0665'   CD 0000*                call    seasym          ;search symbol table
0668'   B7                      or      a               ;found ?
0669'   20 13                   jr      nz,symchk2
066B'   3A 0037"                ld      a,(localf)
066E'   B7                      or      a
066F'   20 05                   jr      nz,symchk1
0671'   CD 1F1A'                call    ilenam
0674'   18 08                   jr      symchk2
0676'                   symchk1:
0676'   3A 0016"                ld      a,(idetyp)
0679'   CB 67                   bit     4,a
067B'   C4 1F1A'                call    nz,ilenam
067E'                   symchk2:
067E'   3A 0037"                ld      a,(localf)
0681'   B7                      or      a
0682'   C8                      ret     z
0683'   3E 10                   ld      a,10h
0685'   C9                      ret
                        ;
```

```
                                ;       **  statement  **
                                ;
0686'                           statem::
0686'   3A 0036"                        ld      a,(tokcod)
0689'   3D                              dec     a               ;identifier ?
068A'   20 41                           jr      nz,statem2
068C'   2A 0000"                        ld      hl,(cptr)
068F'   7E                              ld      a,(hl)
0690'   FE 3A                           cp      ':'             ;label ?
0692'   20 39                           jr      nz,statem2
0694'   23                              inc     hl
0695'   7E                              ld      a,(hl)
0696'   FE 3D                           cp      '='
0698'   28 33                           jr      z,statem2
069A'   22 0000"                        ld      (cptr),hl
069D'   21 0016"                        ld      hl,idetyp
06A0'   CD 0000*                        call    seasym          ;search symbol table
06A3'   B7                              or      a               ;found ?
06A4'   20 16                           jr      nz,stateml
06A6'   3A 0016"                        ld      a,(idetyp)
06A9'   CB 67                           bit     4,a             ;global ?
06AB'   28 0F                           jr      z,stateml
06AD'   FE 98                           cp      98h             ;undefined label ?
06AF'   28 05                           jr      z,statem0
06B1'   CD 1F2C'                        call    ilelab
06B4'   18 06                           jr      stateml
06B6'                           statem0:
06B6'   2A 0023"                        ld      hl,(adrs)
06B9'   CD 19B7'                        call    ptcha           ;put chain address
06BC'                           stateml:
06BC'   2A 000C"                        ld      hl,(cloc)
06BF'   22 0023"                        ld      (adrs),hl
06C2'   21 0016"                        ld      hl,idetyp
06C5'   36 18                           ld      (hl),18h
06C7'   CD 0000*                        call    regsym          ;symbol table register
06CA'   CD 1B0F'                        call    gtoken
06CD'                           statem2:
06CD'   CD 1831'                        call    propt           ;preset operand table pointer
06D0'   3A 0036"                        ld      a,(tokcod)
06D3'   21 06EE'                        ld      hl,stmcod
06D6'   01 000D                         ld      bc,stmcode-stmcod
06D9'   ED B1                           cpir                    ;search
06DB'   C2 0B71'                        jp      nz,idxopr
06DE'   3E 0C                           ld      a,stmcode-stmcod-1
06E0'   91                              sub     c
06E1'   87                              add     a,a
06E2'   4F                              ld      c,a
06E3'   21 06FB'                        ld      hl,stmadr
06E6'   09                              add     hl,bc
06E7'   5E                              ld      e,(hl)
06E8'   23                              inc     hl
06E9'   56                              ld      d,(hl)
06EA'   D5                              push    de
06EB'   C3 1B0F'                        jp      gtoken
                                ;
06EE'   3B 25 7B                stmcod: defb    ';%{'
06F1'   90 96 98 9B                     defb    90h,96h,98h,9bh ;'inline','while','for','loop'
06F5'   93 9E 9D 9C                     defb    93h,9eh,9dh,9ch ;'if','exit','goto','go'
06F9'   9F A0                           defb    9fh,0a0h        ;'return','stop'
06FB'                           stmcode equ     $
                                ;
06FB'   0787' 0715'             stmadr: defw    dumstm,iclstm,comstm,inlstm
06FF'   075C' 05A5'
0703'   0796' 07CC'                     defw    whistm,forstm,lopstm,ifstm
0707'   0902' 09F7'
070B'   0A7B' 0A9B'                     defw    extstm,gotstm,gostm,retstm
070F'   0A93' 0AF1'
0713'   0B16'                           defw    stpstm
                                ;
```

```
                    ; %include statement
                    ;
                    ;       %include d:filename.typ;
0715'               iclstm::
0715'   FE 81               cp      81h             ;include ?
0717'   20 06               jr      nz.brkstm
0719'   CD 0000*            call    inclu           ;include file open
071C'   C3 1B0F'            jp      gtoken
                    ;
                    ; %break statement
                    ;
                    ;       %break;
071F'               brkstm::
071F'   FE 8B               cp      8bh             ;break ?
0721'   20 15               jr      nz,tronst
0723'   CD 1B0F'            call    gtoken
0726'   FE 3B               cp      ';'
0728'   C2 200E'            jp      nz,synerr
072B'   3A 0005"            ld      a,(stmflg)
072E'   B7                  or      a               ;statement compile ?
072F'   CA 200E'            jp      z.synerr
0732'   CD 0000*            call    pbreak          ;put break object
0735'   C3 1B0F'            jp      gtoken
                    ;
                    ; %tron statement
                    ;
                    ;       %tron;
0738'               tronst::
0738'   D6 85               sub     85h             ;tron ?
073A'   20 0E               jr      nz,trofst
073C'   CD 1B0F'            call    gtoken
073F'   FE 3B               cp      ';'
0741'   C2 200E'            jp      nz,synerr
0744'   CD 0000*            call    tron            ;trace on
0747'   C3 1B0F'            jp      gtoken
                    ;
                    ; %troff statement
                    ;
                    ;       %troff;
074A'               trofst::
074A'   3D                  dec     a               ;troff ?
074B'   C2 200E'            jp      nz,synerr
074E'   CD 1B0F'            call    gtoken
0751'   FE 3B               cp      ';'
0753'   C2 200E'            jp      nz.synerr
0756'   CD 0000*            call    troff           ;trace off
0759'   C3 1B0F'            jp      gtoken
                    ;
                    ; compound. until statement
                    ;
                    ;         { statement ... }
075C'               comstm::
075C'   CD 0B45'            call    pshlea
075F'   CD 078B'            call    compoul
                    ;
                    ;         { statement ... } until exp ;

0762'   CD 1B0F'            call    gtoken
0765'   FE 97               cp      97h             ;'until' ?
0767'   20 1B               jr      nz,comstml
0769'   CD 1831'            call    propt
076C'   CD 1B0F'            call    gtoken
076F'   CD 0DAF'            call    expres          ;expression
0772'   3E B7               ld      a.0b7h          ;"  OR   A  "
0774'   CD 1953'            call    ptcodl
0777'   2A 007A"            ld      hl,(lopadr)
077A'   3E 28               ld      a,28h           ;"  JR   Z.LOPADR " or
077C'   CD 0B2A'            call    ptjr
077F'   3E CA               ld      a.0cah          ;"  JP   Z.LOPADR "
```

```
0781'   DC 196E'            call    c,ptcod3
0784'               comstm1:
0784'   CD 0B5D'            call    poplea
0787'               dumstm:
0787'   C9                  ret
                    ;
0788'               compou::
0788'   CD 1B0F'            call    gtoken
078B'               compou1:
078B'   CD 0686'            call    statem          ;statement
078E'   3A 0036"            ld      a,(tokcod)
0791'   FE 7D               cp      '}'
0793'   20 F6               jr      nz,compou1
0795'   C9                  ret
                    ;
                    ; while statement
                    ;
                    ;       while  exp  { statement ... }
0796'               whistm::
0796'   CD 0B45'            call    pshlea
0799'   CD 0DBA'            call    expre           ;expression
079C'   FE 7B               cp      '{'
079E'   C2 200E'            jp      nz,synerr
07A1'   3E B7               ld      a,0b7h          ;"  OR   A  "
07A3'   CD 1953'            call    ptcod1
07A6'   3E CA               ld      a,0cah          ;"  JP   Z,0 "
07A8'   21 0000             ld      hl,0
07AB'   CD 196E'            call    ptcod3
07AE'   2A 000C"            ld      hl,(cloc)
07B1'   2B                  dec     hl
07B2'   2B                  dec     hl
07B3'   22 007C"            ld      (extadr),hl
07B6'   CD 0788'            call    compou          ; {  statement ... }
07B9'   2A 007A"            ld      hl,(lopadr)
07BC'   3E 18               ld      a,18h           ;"  JR   LOPADR  " or
07BE'   CD 0B2A'            call    ptjr
07C1'   3E C3               ld      a,0c3h
07C3'   DC 196E'            call    c,ptcod3        ;"  JP   LOPADR  "
07C6'   CD 0B5D'            call    poplea
07C9'   C3 1B0F'            jp      gtoken
                    ;
                    ; for statement
                    ;
                    ;       for  var := exp_1  to  exp_2  by  exp_3  {  statement ... }
07CC'               forstm::
07CC'   3D                  dec     a               ;identifier ?
07CD'   C2 200E'            jp      nz,synerr
07D0'   21 0016"            ld      hl,idetyp
07D3'   CD 0000*            call    seasym          ;serach symbol table
07D6'   B7                  or      a               ;found ?
07D7'   C4 1E74'            call    nz,misvar
07DA'   21 0016"            ld      hl,idetyp
07DD'   7E                  ld      a,(hl)
07DE'   E6 2F               and     2fh
07E0'   FE 02               cp      2               ;variable name ?
07E2'   C2 200E'            jp      nz,synerr
07E5'   01 000F             ld      bc,adrs-idetyp+2
07E8'   CD 1AF3'            call    blkpsh          ;block push
07EB'   21 0016"            ld      hl,idetyp
07EE'   CB EE               set     5,(hl)
07F0'   CD 0000*            call    regsym          ;symbol table register
07F3'   01 0006             ld      bc,6
07F6'   21 007E"            ld      hl,convar
07F9'   CD 1AF3'            call    blkpsh          ;push convar,terpar,incpar
07FC'   2A 0023"            ld      hl,(adrs)
07FF'   22 007E"            ld      (convar),hl
0802'   CD 1B0F'            call    gtoken
0805'   FE F0               cp      0f0h            ;':=' ?
0807'   C2 200E'            jp      nz,synerr
```

```
080A'   CD 1B0F'          call    gtoken
080D'   CD 0DBA'          call    expre           ;expression_1
0810'   FE 99             cp      99h             ;'to' ?
0812'   C2 200E'          jp      nz,synerr
0815'   3E 32             ld      a,32h           ;"  LD   (CONVAR),A  "
0817'   2A 007E"          ld      hl,(convar)
081A'   CD 196E'          call    ptcod3
081D'   CD 1B0F'          call    gtoken
0820'   CD 0DBA'          call    expre           ;expression_2
0823'   2A 0010"          ld      hl,(wloc)
0826'   E5                push    hl
0827'   22 0080"          ld      (terpar),hl
082A'   CD 08DD'          call    forstm8
082D'   3E 32             ld      a,32h           ;"  LD  (TERPAR),A  "
082F'   2B                dec     hl
0830'   CD 196E'          call    ptcod3
0833'   21 0000           ld      hl,0
0836'   22 0082"          ld      (incpar),hl
0839'   3A 0036"          ld      a,(tokcod)
083C'   FE 7B             cp      '{'
083E'   28 1F             jr      z,forstm1
0840'   FE 9A             cp      9ah             ;'by' ?
0842'   C2 200E'          jp      nz,synerr
0845'   CD 1B0F'          call    gtoken
0848'   CD 0DBA'          call    expre           ;expression_3
084B'   FE 7B             cp      '{'
084D'   C2 200E'          jp      nz,synerr
0850'   2A 0010"          ld      hl,(wloc)
0853'   22 0082"          ld      (incpar).hl
0856'   CD 08DD'          call    forstm8
0859'   3E 32             ld      a,32h           ;"  LD  (INCPAR).A  "
085B'   2B                dec     hl
085C'   CD 196E'          call    ptcod3
085F'             forstm1:
085F'   CD 0B45'          call    pshlea
0862'   2A 0080"          ld      hl.(terpar)
0865'   CD 183D'          call    stoptb          ;opr1 = terminal parameter address
0868'   2A 007E"          ld      hl.(convar)
086B'   CD 183D'          call    stoptb          ;opr2 = control variable address
086E'   21 08EE'          ld      hl,forobj1
0871'   CD 189A'          call    codgen0         ;code ganerate
0874'   2A 000C"          ld      hl.(cloc)
0877'   2B                dec     hl
0878'   2B                dec     hl
0879'   22 007C"          ld      (extadr).hl
087C'   CD 0788'          call    compou          ; {  statement ... }
087F'   2A 0082"          ld      hl.(incpar)
0882'   7C                ld      a.h
0883'   B5                or      l
0884'   20 19             jr      nz.forstm3
0886'   2A 007E"          ld      hl.(convar)
0889'   3E 21             ld      a.21h           ;"  LD   HL.CONVAR  "
088B'   CD 196E'          call    ptcod3
088E'   3E 34             ld      a.34h           ;"  INC  (HL)  "
0890'   CD 1953'          call    ptcod1
0893'   2A 007A"          ld      hl,(lopadr)
0896'   3E 20             ld      a,20h           ;"  JR   NZ.LOPADR  " or
0898'   CD 0B2A'          call    ptjr
089B'   3E C2             ld      a.0c2h          ;"  JP   NZ.LOPADR  "
089D'   18 19             jr      forstm4
089F'             forstm3:
089F'   CD 183D'          call    stoptb          ;opr1 = incrementation parameter address
08A2'   2A 007E"          ld      hl,(convar)
08A5'   CD 183D'          call    stoptb          ;opr2 = control variable address
08A8'   21 08F9'          ld      hl,forobj2
08AB'   CD 189A'          call    codgen0         ;code generate
08AE'   2A 007A"          ld      hl,(lopadr)
08B1'   3E 30             ld      a,30h           ;"  JR   NC.LOPADR  " or
08B3'   CD 0B2A'          call    ptjr
```

```
08B6'  3E D2              ld      a,0d2h          ;"  JP   NC,LOPADR  "
08B8'               forstm4:
08B8'  DC 196E'           call    c,ptcod3
08BB'  CD 0B5D'           call    poplea
08BE'  E1                 pop     hl
08BF'  22 0010"           ld      (wloc),hl
08C2'  01 0006            ld      bc,6
08C5'  21 007E"           ld      hl,convar
08C8'  CD 1B03'           call    blkpop          ;pop incpar,terpar,convar
08CB'  01 000F            ld      bc,adrs-idetyp+2
08CE'  21 0016"           ld      hl,idetyp
08D1'  CD 1B03'           call    blkpop          ;block pop
08D4'  21 0016"           ld      hl,idetyp
08D7'  CD 0000*           call    regsym          ;symbol table register
08DA'  C3 1B0F'           jp      gtoken
                    ;
08DD'               forstm8:
08DD'  23                 inc     hl
08DE'  22 0010"           ld      (wloc),hl
08E1'  EB                 ex      de,hl
08E2'  2A 000E"           ld      hl,(dloc)
08E5'  B7                 or      a
08E6'  ED 52              sbc     hl,de           ; dloc >= wloc ?
08E8'  EB                 ex      de,hl
08E9'  D0                 ret     nc              ;return if so
08EA'  22 000E"           ld      (dloc),hl
08ED'  C9                 ret
                    ;
08EE'  0A           forobj1: defb  10
08EF'  3A B9B9            ld      a,(opr1w)
08F2'  21 BBBB            ld      hl,opr2w
08F5'  96                 sub     (hl)
08F6'  DA 0000            jp      c,0
08F9'  08           forobj2: defb  8
08FA'  3A B9B9            ld      a,(opr1w)
08FD'  21 BBBB            ld      hl,opr2w
0900'  86                 add     a,(hl)
0901'  77                 ld      (hl),a
                    ;
                    ; loop statement
                    ;
                    ;         loop  var, exp  {  statement ... }
0902'               lopstm::
0902'  FE 23              cp      '#'
0904'  CA 098C'           jp      z,lopstb
0907'  3D                 dec     a
0908'  C2 200E'           jp      nz,synerr
090B'  21 0016"           ld      hl,idetyp
090E'  CD 0000*           call    seasym          ;search symbol table
0911'  B7                 or      a               ;found ?
0912'  C4 1E74'           call    nz,misvar
0915'  21 0016"           ld      hl,idetyp
0918'  7E                 ld      a,(hl)
0919'  E6 2F              and     2fh
091B'  FE 02              cp      2               ;variable name ?
091D'  C2 200E'           jp      nz,synerr
0920'  01 000F            ld      bc,adrs-idetyp+2
0923'  CD 1AF3'           call    blkpsh          ;block push
0926'  21 0016"           ld      hl,idetyp
0929'  CB EE              set     5,(hl)
092B'  CD 0000*           call    regsym          ;symbol table register
092E'  2A 007E"           ld      hl,(convar)
0931'  E5                 push    hl
0932'  2A 0023"           ld      hl,(adrs)
0935'  22 007E"           ld      (convar),hl
0938'  CD 1B0F'           call    gtoken
093B'  FE 2C              cp      ','
093D'  C2 200E'           jp      nz,synerr
0940'  CD 1B0F'           call    gtoken
```

```
0943'   CD 0DBA'          call    expre           :expression
0946'   FE 7B             cp      '{'
0948'   C2 200E'          jp      nz.synerr
094B'   3E 32             ld      a.32h           :"  LD  (CONVAR).A "
094D'   2A 007E"          ld      hl.(convar)
0950'   CD 196E'          call    ptcod3
0953'   CD 0B45'          call    pshlea
0956'   CD 0788'          call    compou          : {  statement ... }
0959'   3E 21             ld      a.21h           :"  LD  HL.CONVAR  "
095B'   2A 007E"          ld      hl,(convar)
095E'   CD 196E'          call    ptcod3
0961'   3E 35             ld      a.35h           :"  DEC (HL)  "
0963'   CD 1953'          call    ptcod1
0966'   2A 007A"          ld      hl,(lopadr)
0969'   3E 20             ld      a.20h           :"  JR  NZ.LOPADR " or
096B'   CD 0B2A'          call    ptjr
096E'   3E C2             ld      a.0c2h
0970'   DC 196E'          call    c,ptcod3        :"  JP  NZ.LOPADR "
0973'   CD 0B5D'          call    poplea
0976'   E1                pop     hl
0977'   22 007E"          ld      (convar).hl
097A'   01 000F           ld      bc.adrs-idetyp+2
097D'   21 0016"          ld      hl.idetyp
0980'   CD 1B03'          call    blkpop          ;block pop
0983'   21 0016"          ld      hl,idetyp
0986'   CD 0000*          call    regsym          ;symbol table register
0989'   C3 1B0F'          jp      gtoken
                  ;
                  ;          loop #, exp  {  statement ... }
098C'             lopstb:
098C'   21 0039"          ld      hl,loopb
098F'   7E                ld      a,(hl)
0990'   B7                or      a
0991'   C2 200E'          jp      nz,synerr
0994'   2F                cpl
0995'   77                ld      (hl),a
0996'   CD 1B0F'          call    gtoken
0999'   FE 2C             cp      ','
099B'   C2 200E'          jp      nz,synerr
099E'   CD 1B0F'          call    gtoken
09A1'   CD 0DF5'          call    logexp          ;expression
09A4'   47                ld      b,a
09A5'   3A 003B"          ld      a,(optble)
09A8'   3D                dec     a
09A9'   C2 203D'          jp      nz,experr
09AC'   78                ld      a,b
09AD'   FE 7B             cp      '{'
09AF'   C2 200E'          jp      nz,synerr
09B2'   FD 7E FD          ld      a,(iy-3)
09B5'   B7                or      a
09B6'   28 11             jr      z,lopstb2
09B8'   21 0DEC'          ld      hl,ldexa
09BB'   3D                dec     a
09BC'   28 08             jr      z,lopstb1
09BE'   21 09F4'          ld      hl,ldbim
09C1'   CD 188F'          call    codgen          ;code generate
09C4'   18 08             jr      lopstb3
09C6'             lopstb1:
09C6'   CD 188F'          call    codgen          ;code generate
09C9'             lopstb2:
09C9'   3E 47             ld      a,47h           ;"  LD  B,A  "
09CB'   CD 1953'          call    ptcod1
09CE'             lopstb3:
09CE'   CD 0B45'          call    pshlea
09D1'   CD 0788'          call    compou          ; {  statement ... }
09D4'   2A 007A"          ld      hl,(lopadr)
09D7'   3E 10             ld      a,10h           ;"  DJNZ LOPADR  " or
09D9'   CD 0B2A'          call    ptjr
09DC'   30 0C             jr      nc,lopstb4
```

```
09DE'   E5                      push    hl
09DF'   3E 05                   ld      a,05h           ;"  DEC  B  "
09E1'   CD 1953'                call    ptcod1
09E4'   E1                      pop     hl
09E5'   3E C2                   ld      a,0c2h          ;"  JP   NZ,LOPADR "
09E7'   CD 196E'                call    ptcod3
09EA'                   lopstb4:
09EA'   AF                      xor     a
09EB'   32 0039"                ld      (loopb),a
09EE'   CD 0B5D'                call    poplea
09F1'   C3 1B0F'                jp      gtoken
                        ;
09F4'   02              ldbim:  defb    2
09F5'   06 B8                   ld      b,opr1b
                        ;
                        ; if statement
                        ;
                        ;           if   exp  then  statement_1  else  statement_2
09F7'                   ifstm::
09F7'   2A 0086"                ld      hl,(nxtadr)
09FA'   E5                      push    hl
09FB'   2A 0084"                ld      hl,(skpadr)
09FE'   E5                      push    hl
09FF'   21 0000                 ld      hl,0
0A02'   22 0086"                ld      (nxtadr),hl
0A05'   22 0084"                ld      (skpadr),hl
0A08'                   ifstm1:
0A08'   CD 0DBA'                call    expre           ;expression
0A0B'   FE 94                   cp      94h             ;'then' ?
0A0D'   C2 200E'                jp      nz,synerr
0A10'   3E B7                   ld      a,0b7h          ;"  OR  A  "
0A12'   CD 1953'                call    ptcod1
0A15'   3E CA                   ld      a,0cah          ;"  JP  Z,0 "
0A17'   21 0000                 ld      hl,0
0A1A'   CD 196E'                call    ptcod3
0A1D'   2A 000C"                ld      hl,(cloc)
0A20'   2B                      dec     hl
0A21'   2B                      dec     hl
0A22'   22 0084"                ld      (skpadr),hl
0A25'   CD 1B0F'                call    gtoken
0A28'   CD 0686'                call    statem          ;statement_1
0A2B'   3A 0036"                ld      a,(tokcod)
0A2E'   FE BD                   cp      0bdh            ;'elseif' ?
0A30'   28 2A                   jr      z,ifstm4
0A32'   FE 95                   cp      095h            ;'else' ?
0A34'   28 08                   jr      z,ifstm2
0A36'   2A 0084"                ld      hl,(skpadr)
0A39'   CD 19B7'                call    ptcha           ;put chain address
0A3C'   18 0D                   jr      ifstm3
0A3E'                   ifstm2:
0A3E'   CD 1B0F'                call    gtoken
0A41'   FE 93                   cp      93h             ;'if' ?
0A43'   28 17                   jr      z,ifstm4
0A45'   CD 0A64'                call    ifstm5
0A48'   CD 0686'                call    statem          ;statement_2
0A4B'                   ifstm3:
0A4B'   2A 0086"                ld      hl,(nxtadr)
0A4E'   7C                      ld      a,h
0A4F'   B5                      or      l
0A50'   C4 19B7'                call    nz,ptcha        ;put chain address
0A53'   E1                      pop     hl
0A54'   22 0084"                ld      (skpadr),hl
0A57'   E1                      pop     hl
0A58'   22 0086"                ld      (nxtadr),hl
0A5B'   C9                      ret
0A5C'                   ifstm4:
0A5C'   CD 0A64'                call    ifstm5
0A5F'   CD 1B0F'                call    gtoken
0A62'   18 A4                   jr      ifstm1
```

```
                         ;
0A64'                    ifstm5:
0A64'   2A 0086"                 ld      hl,(nxtadr)
0A67'   3E C3                    ld      a,0c3h          ;"  JP  NXTADR  "
0A69'   CD 196E'                 call    ptcod3
0A6C'   2A 0084"                 ld      hl,(skpadr)
0A6F'   CD 19B7'                 call    ptcha           ;put chain address
0A72'   2A 000C"                 ld      hl,(cloc)
0A75'   2B                       dec     hl
0A76'   2B                       dec     hl
0A77'   22 0086"                 ld      (nxtadr),hl
0A7A'   C9                       ret
                         ;
                         ; exit statement
                         ;
                         ;       exit ;
0A7B'                    extstm::
0A7B'   FE 3B                    cp      ';'
0A7D'   C2 200E'                 jp      nz,synerr
0A80'   3E C3                    ld      a,0c3h          ;"  JP  EXTADR  "
0A82'   2A 007C"                 ld      hl,(extadr)
0A85'   CD 196E'                 call    ptcod3
0A88'   2A 000C"                 ld      hl,(cloc)
0A8B'   2B                       dec     hl
0A8C'   2B                       dec     hl
0A8D'   22 007C"                 ld      (extadr),hl
0A90'   C3 1B0F'                 jp      gtoken
                         ;
                         ; goto statement
                         ;
                         ;       goto label;    go to label;
0A93'                    gostm::
0A93'   FE 99                    cp      99h             ;'to' ?
0A95'   C2 200E'                 jp      nz,synerr
0A98'   CD 1B0F'                 call    gtoken
0A9B'                    gotstm::
0A9B'   3D                       dec     a
0A9C'   C2 200E'                 jp      nz,synerr
0A9F'   21 0016"                 ld      hl,idetyp
0AA2'   CD 0000*                 call    seasym          ;search symbol table
0AA5'   B7                       or      a               ;found ?
0AA6'   28 0D                    jr      z,gotstm1
0AA8'   3E 98                    ld      a,98h
0AAA'   32 0016"                 ld      (idetyp),a
0AAD'   21 0000                  ld      hl,0
0AB0'   22 0023"                 ld      (adrs),hl
0AB3'   18 1E                    jr      gotstm2
0AB5'                    gotstm1:
0AB5'   3A 0016"                 ld      a,(idetyp)
0AB8'   47                       ld      b,a
0AB9'   E6 0F                    and     0fh
0ABB'   FE 08                    cp      8               ;label ?
0ABD'   C2 200E'                 jp      nz,synerr
0AC0'   2A 0023"                 ld      hl,(adrs)
0AC3'   78                       ld      a,b
0AC4'   87                       add     a,a             ;defined label ?
0AC5'   38 0C                    jr      c,gotstm2
0AC7'   3E 18                    ld      a,18h           ;"  JR  LABEL  " or
0AC9'   CD 0B2A'                 call    ptjr
0ACC'   3E C3                    ld      a,0c3h          ;"  JP  LABEL  "
0ACE'   DC 196E'                 call    c,ptcod3
0AD1'   18 13                    jr      gotstm3
0AD3'                    gotstm2:
0AD3'   3E C3                    ld      a,0c3h          ;"  JP  LABEL  "
0AD5'   CD 196E'                 call    ptcod3
0AD8'   2A 000C"                 ld      hl,(cloc)
0ADB'   2B                       dec     hl
0ADC'   2B                       dec     hl
0ADD'   22 0023"                 ld      (adrs),hl
```

```
0AE0'   21 0016"            ld      hl,idetyp
0AE3'   CD 0000*            call    regsym          ;symbol table register
0AE6'               gotstm3:
0AE6'   CD 1B0F'            call    gtoken
0AE9'   FE 3B               cp      ';'
0AEB'   C2 200E'            jp      nz,synerr
0AEE'   C3 1B0F'            jp      gtoken
                    ;
                    ; return statement
                    ;
                    ;       return ;
0AF1'               retstm::
0AF1'   FE 3B               cp      ';'
0AF3'   C2 200E'            jp      nz,synerr
0AF6'   3A 008C"            ld      a,(idxpsf)
0AF9'   B7                  or      a               ;index push ?
0AFA'   28 12               jr      z,retstm1
0AFC'   3E C3               ld      a,0c3h          ;"  JP  RETJAD  "
0AFE'   2A 008D"            ld      hl,(retjad)
0B01'   CD 196E'            call    ptcod3
0B04'   2A 000C"            ld      hl,(cloc)
0B07'   2B                  dec     hl
0B08'   2B                  dec     hl
0B09'   22 008D"            ld      (retjad),hl
0B0C'   18 05               jr      retstm2
0B0E'               retstm1:
0B0E'   3E C9               ld      a,0c9h          ;"  RET  "
0B10'   CD 1953'            call    ptcod1
0B13'               retstm2:
0B13'   C3 1B0F'            jp      gtoken
                    ;
                    ; stop statement
                    ;
                    ;       stop ;
0B16'               stpstm::
0B16'   FE 3B               cp      ';'
0B18'   C2 200E'            jp      nz,synerr
0B1B'   3E C3               ld      a,0c3h          ;"  JP  @STOP "
0B1D'   2A 0006"            ld      hl,(codorg)
0B20'   11 0000*            ld      de,@stop-@start
0B23'   19                  add     hl,de
0B24'   CD 196E'            call    ptcod3
0B27'   C3 1B0F'            jp      gtoken
                    ;
                    ;       **  put relative jump  **
                    ;
0B2A'               ptjr::  :       a : op code
                            :       hl: jump address
0B2A'   E5                  push    hl
0B2B'   F5                  push    af
0B2C'   ED 5B 000C"         ld      de,(cloc)
0B30'   13                  inc     de
0B31'   13                  inc     de
0B32'   B7                  or      a
0B33'   ED 52               sbc     hl,de           ; hl := hl - $ + 2
0B35'   D1                  pop     de
0B36'   5D                  ld      e,l
0B37'   7D                  ld      a,l
0B38'   87                  add     a,a
0B39'   9F                  sbc     a,a
0B3A'   BC                  cp      h
0B3B'   E1                  pop     hl
0B3C'   20 05               jr      nz,ptjr1
0B3E'   CD 1964'            call    ptcod2
0B41'   B7                  or      a               ; cy=0 : OK
0B42'   C9                  ret
0B43'               ptjr1:
0B43'   37                  scf                     ; cy=1 : NG
0B44'   C9                  ret
```

```
                         ;
                         ;       **  push loop address, exit jump address & preset   **
                         ;
0B45'                    pshlea::
0B45'   DD E1                    pop     ix
0B47'   2A 007A"                 ld      hl,(lopadr)
0B4A'   E5                       push    hl
0B4B'   2A 007C"                 ld      hl,(extadr)
0B4E'   E5                       push    hl
0B4F'   21 0000                  ld      hl,0
0B52'   22 007C"                 ld      (extadr),hl
0B55'   2A 000C"                 ld      hl,(cloc)
0B58'   22 007A"                 ld      (lopadr),hl
0B5B'   DD E9                    jp      (ix)
                         ;
                         ;       **  pop loop address, exit jump address   **
                         ;
0B5D'                    poplea::
0B5D'   DD E1                    pop     ix
0B5F'   2A 007C"                 ld      hl,(extadr)
0B62'   7C                       ld      a,h
0B63'   B5                       or      l
0B64'   C4 19B7'                 call    nz,ptcha        ;put chain address
0B67'   E1                       pop     hl
0B68'   22 007C"                 ld      (extadr),hl
0B6B'   E1                       pop     hl
0B6C'   22 007A"                 ld      (lopadr),hl
0B6F'   DD E9                    jp      (ix)
                         ;
                         ;       **  index operation  **
                         ;
0B71'                    idxopr::
0B71'   3A 0036"                 ld      a,(tokcod)
0B74'   D6 A1                    sub     0a1h       ;'set' or 'ldx' or 'stx' or 'inx' or 'dex' ?
0B76'   DA 0DAF'                 jp      c,expres
0B79'   FE 05                    cp      5
0B7B'   D2 0DAF'                 jp      nc,expres
0B7E'   87                       add     a,a
0B7F'   5F                       ld      e,a
0B80'   16 00                    ld      d,0
0B82'   21 0B8D'                 ld      hl,idopad
0B85'   19                       add     hl,de
0B86'   5E                       ld      e,(hl)
0B87'   23                       inc     hl
0B88'   56                       ld      d,(hl)
0B89'   D5                       push    de
0B8A'   C3 1B0F'                 jp      gtoken
                         ;
0B8D'   0B97' 0C44'      idopad: defw    idxset,ldx,stx,inx,dex
0B91'   0CED' 0D9D'
0B95'   0DA0'
                         ;
                         ;           set  index := const +- expression;
                         ;           set  index := index +- expression;
0B97'                    idxset::
0B97'   D6 BF                    sub     0bfh            ;'ix' ?
0B99'   28 05                    jr      z,idxset1
0B9B'   FE 01                    cp      1               ;'iy' ?
0B9D'   C2 2022'                 jp      nz,idxerr
0BA0'                    idxset1:
0BA0'   F5                       push    af
0BA1'   CD 1B0F'                 call    gtoken
0BA4'   FE F0                    cp      0f0h            ;':=' ?
0BA6'   C2 2022'                 jp      nz,idxerr
0BA9'   CD 1B0F'                 call    gtoken
0BAC'   D6 BF                    sub     0bfh            ;'ix' ?
0BAE'   28 0B                    jr      z,idxset2
0BB0'   FE 01                    cp      1               ;'iy' ?
0BB2'   28 07                    jr      z,idxset2
```

```
0BB4'   CD 1A01'              call    wodcon
0BB7'   3E 02                 ld      a,2
0BB9'   18 05                 jr      idxset3
0BBB'                 idxset2:
0BBB'   F5                    push    af
0BBC'   CD 1B0F'              call    gtoken
0BBF'   F1                    pop     af
0BC0'                 idxset3:
0BC0'   D1                    pop     de
0BC1'   5F                    ld      e,a
0BC2'   3A 0036"              ld      a,(tokcod)
0BC5'   FE 3B                 cp      ';'
0BC7'   28 43                 jr      z,idxset7
0BC9'   FE 2B                 cp      '+'
0BCB'   28 05                 jr      z,idxset4
0BCD'   D6 2D                 sub     '-'
0BCF'   C2 2022'              jp      nz,idxerr
0BD2'                 idxset4:
0BD2'   F5                    push    af
0BD3'   D5                    push    de
0BD4'   E5                    push    hl
0BD5'   CD 1B0F'              call    gtoken
0BD8'   CD 0DAF'              call    expres          ;expression
0BDB'   E1                    pop     hl
0BDC'   D1                    pop     de
0BDD'   D5                    push    de
0BDE'   CD 0C12'              call    idxset10
0BE1'   D1                    pop     de
0BE2'   15                    dec     d
0BE3'   16 DD                 ld      d,0ddh          ;index ix
0BE5'   20 02                 jr      nz,$+4
0BE7'   16 FD                   ld    d,0fdh          ;index iy
0BE9'   1E 19                 ld      e,19h
0BEB'   F1                    pop     af
0BEC'   D5                    push    de
0BED'   B7                    or      a
0BEE'   28 0A                 jr      z,idxset5
0BF0'   3E 5F                 ld      a,5fh           ;"  LD   E,A  "
0BF2'   21 0016               ld      hl,16h          ;"  LD   D,0  "
0BF5'   CD 196E'              call    ptcod3
0BF8'   18 0E                 jr      idxset6
0BFA'                 idxset5:
0BFA'   11 2F5F               ld      de,2f5fh        ;"  CPL        "
0BFD'   CD 1964'              call    ptcod2          ;"  LD   E,A  "
0C00'   3E 16                 ld      a,16h           ;"  LD   D,0FFH "
0C02'   21 13FF               ld      hl,13ffh        ;"  INC  DE   "
0C05'   CD 196E'              call    ptcod3
0C08'                 idxset6:
0C08'   D1                    pop     de              ;"  ADD  IX,DE " or "  ADD  IY,DE "
0C09'   C3 1964'              jp      ptcod2
0C0C'                 idxset7:
0C0C'   CD 0C12'              call    idxset10
0C0F'   C3 1B0F'              jp      gtoken
                      ;
0C12'                 idxset10:
0C12'   7B                    ld      a,e
0C13'   BA                    cp      d
0C14'   C8                    ret     z
0C15'   B7                    or      a               ;' iy := ix ' ?
0C16'   20 0C                 jr      nz,idxset11
0C18'   11 DDE5               ld      de,0dde5h       ;"  PUSH IX  "
0C1B'   CD 1964'              call    ptcod2
0C1E'   11 FDE1               ld      de,0fde1h       ;"  POP  IY  "
0C21'   C3 1964'              jp      ptcod2
0C24'                 idxset11:
0C24'   3D                    dec     a               ;' ix := iy ' ?
0C25'   20 0C                 jr      nz,idxset12
0C27'   11 FDE5               ld      de,0fde5h       ;"  PUSH IY  "
0C2A'   CD 1964'              call    ptcod2
```

```
0C2D'   11 DDE1                 ld      de,0dde1h       ;"  POP  IX  "
0C30'   C3 1964'                jp      ptcod2
0C33'                   idxset12:
0C33'   15                      dec     d
0C34'   3E DD                   ld      a,0ddh
0C36'   20 02                   jr      nz,$+4
0C38'   3E FD                     ld    a,0fdh
0C3A'   E5                      push    hl
0C3B'   CD 1953'                call    ptcod1
0C3E'   E1                      pop     hl
0C3F'   3E 21                   ld      a,21h           ;"  LD   IX,nn " or "  LD   IY,nn "
0C41'   C3 196E'                jp      ptcod3
                        ;
                        ;           ldx  index := var[ exp ];
0C44'                   ldx::
0C44'   16 DD                   ld      d,0ddh
0C46'   D6 BF                   sub     0bfh            ;'ix' ?
0C48'   28 06                   jr      z,ldx1
0C4A'   16 FD                   ld      d,0fdh
0C4C'   3D                      dec     a               ;'iy' ?
0C4D'   C2 2022'                jp      nz,idxerr
0C50'                   ldx1:
0C50'   D5                      push    de
0C51'   CD 1B0F'                call    gtoken
0C54'   FE F0                   cp      0f0h            ;':=' ?
0C56'   C2 2022'                jp      nz,idxerr
0C59'   CD 1B0F'                call    gtoken
0C5C'   3D                      dec     a
0C5D'   C2 2022'                jp      nz,idxerr
0C60'   21 0016"                ld      hl,idetyp
0C63'   CD 0000*                call    seasym          ;search symbol table
0C66'   B7                      or      a               ;found ?
0C67'   C4 1E74'                call    nz,misvar
0C6A'   2A 0023"                ld      hl,(adrs)
0C6D'   3A 0016"                ld      a,(idetyp)
0C70'   E6 0F                   and     0fh
0C72'   D6 02                   sub     2               ;variable name ?
0C74'   28 04                   jr      z,ldx2
0C76'   3D                      dec     a               ;data name ?
0C77'   C2 2022'                jp      nz,idxerr
0C7A'                   ldx2:
0C7A'   E5                      push    hl
0C7B'   CD 1B0F'                call    gtoken
0C7E'   FE 5B                   cp      '['
0C80'   20 44                   jr      nz,ldx6
0C82'   CD 1B0F'                call    gtoken
0C85'   CD 0DF5'                call    logexp          ;expression
0C88'   3A 003B"                ld      a,(optble)
0C8B'   3D                      dec     a
0C8C'   C2 203D'                jp      nz,experr
0C8F'   3A 0036"                ld      a,(tokcod)
0C92'   FE 5D                   cp      ']'
0C94'   C2 2022'                jp      nz,idxerr
0C97'   CD 1B0F'                call    gtoken
0C9A'   FD 7E FD                ld      a,(iy-3)
0C9D'   FE 02                   cp      2               ;constant ?
0C9F'   28 1B                   jr      z,ldx5
0CA1'   B7                      or      a               ;expression ?
0CA2'   28 06                   jr      z,ldx3
0CA4'   21 0CDD'                ld      hl,ldxobj1
0CA7'   CD 188F'                call    codgen          ;code generate
0CAA'                   ldx3:
0CAA'   E1                      pop     hl
0CAB'   CD 183D'                call    stoptb          ;store operand table
0CAE'   21 0CE1'                ld      hl,ldxobj2
0CB1'   CD 188F'                call    codgen
0CB4'   D1                      pop     de
0CB5'   1E E1                   ld      e,0e1h          ;"  POP  IX " or "  POP  IY "
0CB7'   CD 1964'                call    ptcod2
```

```
0CBA'   18 16             jr      ldx7
0CBC'             ldx5:
0CBC'   E1                pop     hl
0CBD'   FD 7E FE          ld      a,(iy-2)
0CC0'   85                add     a,l
0CC1'   6F                ld      l,a
0CC2'   30 01             jr      nc,$+3
0CC4'   24                  inc   h              ; hl := hl + opr
0CC5'   FE                defb    0feh           ; skip 1 byte
0CC6'             ldx6:
0CC6'   E1                pop     hl
0CC7'   E3                ex      (sp),hl
0CC8'   EB                ex      de,hl
0CC9'   1E 2A             ld      e,2ah          ;"  LD  IX,(nn) " or "  LD  IY,(nn) "
0CCB'   CD 1964'          call    ptcod2
0CCE'   E1                pop     hl
0CCF'   CD 1966'          call    ptcodw
0CD2'             ldx7:
0CD2'   3A 0036"          ld      a,(tokcod)
0CD5'   FE 3B             cp      ';'
0CD7'   CA 1B0F'          jp      z,gtoken
0CDA'   C3 2022'          jp      idxerr         ;error : bad index operation
                  ;
0CDD'   03        ldxobj1: defb   3
0CDE'   3A B9B9           ld      a,(oprlw)
                  ;
0CE1'   0B        ldxobj2: defb   11
0CE2'   5F                ld      e,a
0CE3'   16 00             ld      d,0
0CE5'   21 B9B9           ld      hl,oprlw
0CE8'   19                add     hl,de
0CE9'   5E                ld      e,(hl)
0CEA'   23                inc     hl
0CEB'   56                ld      d,(hl)
0CEC'   D5                push    de
                  ;
                  ;        stx  var[ exp ] := index;
0CED'             stx::
0CED'   3D                dec     a
0CEE'   C2 2022'          jp      nz,idxerr
0CF1'   21 0016"          ld      hl,idetyp
0CF4'   CD 0000*          call    seasym         ;search symbol table
0CF7'   B7                or      a              ;found ?
0CF8'   C4 1E74'          call    nz,misvar
0CFB'   2A 0023"          ld      hl,(adrs)
0CFE'   3A 0016"          ld      a,(idetyp)
0D01'   E6 0F             and     0fh
0D03'   FE 02             cp      2              ;variable name ?
0D05'   C2 2022'          jp      nz,idxerr
0D08'   E5                push    hl
0D09'   CD 1B0F'          call    gtoken
0D0C'   FE 5B             cp      '['
0D0E'   20 5C             jr      nz,stx3
0D10'   CD 1B0F'          call    gtoken
0D13'   CD 0DF5'          call    logexp         ;expression
0D16'   3A 003B"          ld      a,(optble)
0D19'   3D                dec     a
0D1A'   C2 203D'          jp      nz,experr
0D1D'   3A 0036"          ld      a,(tokcod)
0D20'   FE 5D             cp      ']'
0D22'   C2 2022'          jp      nz,idxerr
0D25'   FD 7E FD          ld      a,(iy-3)
0D28'   FE 02             cp      2              ;constant ?
0D2A'   28 33             jr      z,stx2
0D2C'   B7                or      a              ;expression ?
0D2D'   28 06             jr      z,stx1
0D2F'   21 0CDD'          ld      hl,ldxobj1
0D32'   CD 188F'          call    codgen         ;code generate
0D35'             stx1:
```

```
0D35'   E1                      pop     hl
0D36'   CD 183D'                call    stoptb          ;store operand table
0D39'   21 0D90'                ld      hl,stxobj1
0D3C'   CD 188F'                call    codgen
0D3F'   CD 1B0F'                call    gtoken
0D42'   FE F0                   cp      0f0h            ;':=' ?
0D44'   C2 2022'                jp      nz,idxerr
0D47'   CD 1B0F'                call    gtoken
0D4A'   16 DD                   ld      d,0ddh          ;index ix
0D4C'   FE BF                   cp      0bfh
0D4E'   28 02                   jr      z,$+4
0D50'   16 FD                     ld    d,0fdh          ;index iy
0D52'   1E E5                   ld      e,0e5h          ;" PUSH IX " or " PUSH IY "
0D54'   CD 1964'                call    ptcod2
0D57'   21 0D98'                ld      hl,stxobj2
0D5A'   CD 188F'                call    codgen
0D5D'   18 26                   jr      stx4
0D5F'                   stx2:
0D5F'   E1                      pop     hl
0D60'   FD 7E FE                ld      a,(iy-2)
0D63'   85                      add     a,l
0D64'   6F                      ld      l,a
0D65'   30 01                   jr      nc,$+3
0D67'   24                        inc   h               ; hl := hl + opr
0D68'   E5                      push    hl
0D69'   CD 1B0F'                call    gtoken
0D6C'                   stx3:
0D6C'   FE F0                   cp      0f0h            ;':=' ?
0D6E'   C2 2022'                jp      nz,idxerr
0D71'   CD 1B0F'                call    gtoken
0D74'   16 DD                   ld      d,0ddh          ;index ix
0D76'   FE BF                   cp      0bfh
0D78'   28 02                   jr      z,$+4
0D7A'   16 FD                     ld    d,0fdh          ;index iy
0D7C'   1E 22                   ld      e,22h
0D7E'   CD 1964'                call    ptcod2
0D81'   E1                      pop     hl
0D82'   CD 1966'                call    ptcodw
0D85'                   stx4:
0D85'   CD 1B0F'                call    gtoken
0D88'   FE 3B                   cp      ';'
0D8A'   CA 1B0F'                jp      z,gtoken
0D8D'   C3 2022'                jp      idxerr
                        ;
0D90'   07              stxobj1: defb   7
0D91'   5F                      ld      e,a
0D92'   16 00                   ld      d,0
0D94'   21 B9B9                 ld      hl,oprlw
0D97'   19                      add     hl,de
                        ;
0D98'   04              stxobj2: defb   4
0D99'   D1                      pop     de
0D9A'   73                      ld      (hl),e
0D9B'   23                      inc     hl
0D9C'   72                      ld      (hl),d
                        ;
                        ;       inx  index:
0D9D'                   inx::
0D9D'   1E 23                   ld      e,23h
0D9F'   21                      defb    21h             ;skip 2 byte
                        ;
                        ;       dex  index:
0DA0'                   dex::
0DA0'   1E 2B                   ld      e,2bh
0DA2'   16 DD                   ld      d,0ddh          ;index ix
0DA4'   FE BF                   cp      0bfh
0DA6'   28 02                   jr      z,$+4
0DA8'   16 FD                     ld    d,0fdh          ;index iy
0DAA'   CD 1964'                call    ptcod2
```

```
0DAD'   18 D6                   jr      stx4
                        ;
                        ;       **  expression  **
                        ;
0DAF'                   expres::
0DAF'   CD 0DBD'                call    expr0
0DB2'   FE 3B                   cp      ';'
0DB4'   CA 1B0F'                jp      z,gtoken
0DB7'   C3 203D'                jp      experr          ;error : bad expression
                        ;
0DBA'                   expre::
0DBA'   CD 1831'                call    propt           ;preset operand table pointer
0DBD'                   expr0:
0DBD'   CD 0DCF'                call    expr2           ;make expression object
0DC0'   3A 003B"                ld      a,(optble)      ;check
0DC3'   3D                      dec     a
0DC4'   C2 203D'                jp      nz,experr
0DC7'   3A 0036"                ld      a,(tokcod)
0DCA'   C9                      ret
                        ;
0DCB'                   expr1:
0DCB'   AF                      xor     a
0DCC'   32 003A"                ld      (puhptf),a
0DCF'                   expr2:
0DCF'   CD 0DF5'                call    logexp          ;logical expression
                        ;
0DD2'                   expr5:
0DD2'   FD 7E FD                ld      a,(iy-3)
0DD5'   B7                      or      a
0DD6'   C8                      ret     z
0DD7'   21 0DEC'                ld      hl,ldexa
0DDA'   3D                      dec     a
0DDB'   28 0C                   jr      z,expr6
0DDD'   21 0DF3'                ld      hl,xora
0DE0'   FD 7E FE                ld      a,(iy-2)
0DE3'   B7                      or      a               ;zero ?
0DE4'   28 03                   jr      z,expr6
0DE6'   21 0DF0'                ld      hl,ldime
0DE9'                   expr6:
0DE9'   C3 188F'                jp      codgen          ;code generate
                        ;
0DEC'   FD              ldexa:  defb    -3
0DED'   3A B9B9                 ld      a,(opr1w)
0DF0'   FE              ldime:  defb    -2
0DF1'   3E B8                   ld      a,opr1b
0DF3'   FF              xora:   defb    -1
0DF4'   AF                      xor     a
                        ;
                        ;  logical expression
                        ;
0DF5'                   logexp::
0DF5'   CD 0EA7'                call    logtem0         ;logical term
0DF8'                   logexp1:
0DF8'   3A 0036"                ld      a,(tokcod)
0DFB'   FE A6                   cp      0a6h            ;'xor' ?
0DFD'   28 21                   jr      z,logexp6
0DFF'   FE A7                   cp      0a7h            ;'or' ?
0E01'   28 07                   jr      z,logexp4
0E03'   FE 21                   cp      '!'
0E05'   28 03                   jr      z,logexp4
0E07'   FE 7C                   cp      '|'
0E09'   C0                      ret     nz
                        ;
                        ;       opr1 := opr1 | opr2
0E0A'                   logexp4:
0E0A'   CD 0EA4'                call    logtem          ;logical term
0E0D'   21 0E36'                ld      hl,orobj
0E10'   CD 1876'                call    dyagen          ;code generate
0E13'   30 E3                   jr      nc,logexp1
```

```
0E15'   FD 7E FE                ld      a,(iy-2)
0E18'   FD B6 01                or      (iy+1)
0E1B'   FD 77 FE                ld      (iy-2),a        ; m(iy-2) := m(iy-2) | m(iy+1)
0E1E'   18 D8                   jr      logexp1
                        ;
                        ;       opr1 := opr1 xor opr2
0E20'                   logexp6:
0E20'   CD 0EA4'                call    logtem
0E23'   21 0E6D'                ld      hl,xorobj
0E26'   CD 1876'                call    dyagen          ;code generate
0E29'   30 CD                   jr      nc,logexp1
0E2B'   FD 7E FE                ld      a,(iy-2)
0E2E'   FD AE 01                xor     (iy+1)
0E31'   FD 77 FE                ld      (iy-2),a        ; m(iy-2) := m(iy-2) xor m(iy+1)
0E34'   18 C2                   jr      logexp1
                        ;
                        ; --  or object  --
                        ;
0E36'   0E46' 0E49'     orobj:  defw    oree,orev,orec
0E3A'   0E4E'
0E3C'   0E51' 0E56'             defw    orve,orvv,orvc
0E40'   0E5E'
0E42'   0E64' 0E67'             defw    orce,orcv

0E46'   02              oree:   defb    2
0E47'   D1                      pop     de
0E48'   B2                      or      d

0E49'   04              orev:   defb    4
0E4A'   21 BBBB                 ld      hl,opr2w
0E4D'   B6                      or      (hl)

0E4E'   02              orec:   defb    2
0E4F'   F6 BA                   or      opr2b

0E51'   04              orve:   defb    4
0E52'   21 B9B9                 ld      hl,opr1w
0E55'   B6                      or      (hl)

0E56'   F9              orvv:   defb    -7
0E57'   3A B9B9                 ld      a,(opr1w)
0E5A'   21 BBBB                 ld      hl,opr2w
0E5D'   B6                      or      (hl)

0E5E'   FB              orvc:   defb    -5
0E5F'   3A B9B9                 ld      a,(opr1w)
0E62'   F6 BA                   or      opr2b

0E64'   02              orce:   defb    2
0E65'   F6 B8                   or      opr1b

0E67'   FB              orcv:   defb    -5
0E68'   3A BBBB                 ld      a,(opr2w)
0E6B'   F6 B8                   or      opr1b
                        ;
                        ; --  exclusive or object  --
                        ;
0E6D'   0E7D' 0E80'     xorobj: defw    xoree,xorev,xorec
0E71'   0E85'
0E73'   0E88' 0E8D'             defw    xorve,xorvv,xorvc
0E77'   0E95'
0E79'   0E9B' 0E9E'             defw    xorce,xorcv

0E7D'   02              xoree:  defb    2
0E7E'   D1                      pop     de
0E7F'   AA                      xor     d

0E80'   04              xorev:  defb    4
```

```
0E81'  21 BBBB              ld      hl,opr2w
0E84'  AE                   xor     (hl)

0E85'  02          xorec:   defb    2
0E86'  EE BA                xor     opr2b

0E88'  04          xorve:   defb    4
0E89'  21 B9B9              ld      hl,opr1w
0E8C'  AE                   xor     (hl)

0E8D'  F9          xorvv:   defb    -7
0E8E'  3A B9B9              ld      a,(opr1w)
0E91'  21 BBBB              ld      hl,opr2w
0E94'  AE                   xor     (hl)

0E95'  FB          xorvc:   defb    -5
0E96'  3A B9B9              ld      a,(opr1w)
0E99'  EE BA                xor     opr2b

0E9B'  02          xorce:   defb    2
0E9C'  EE B8                xor     opr1b

0E9E'  FB          xorcv:   defb    -5
0E9F'  3A BBBB              ld      a,(opr2w)
0EA2'  EE B8                xor     opr1b

                   ;
                   ;  logical term
                   ;
0EA4'              logtem::
0EA4'  CD 1B0F'             call    gtoken          ;get next token
0EA7'              logtem0:
0EA7'  CD 0F04'             call    logfac0         ;logical factor
0EAA'              logtem1:
0EAA'  3A 0036"             ld      a,(tokcod)
0EAD'  FE A8                cp      0a8h            ;'and' ?
0EAF'  28 03                jr      z,logtem2
0EB1'  FE 26                cp      '&'
0EB3'  C0                   ret     nz
                   ;
                   ;        opr1 := opr1 & opr2
0EB4'              logtem2:
0EB4'  CD 0F01'             call    logfac          ;logical factor
0EB7'  21 0ECA'             ld      hl,andobj
0EBA'  CD 1876'             call    dyagen          ;code generate
0EBD'  30 EB                jr      nc,logtem1
0EBF'  FD 7E FE             ld      a,(iy-2)
0EC2'  FD A6 01             and     (iy+1)
0EC5'  FD 77 FE             ld      (iy-2),a        ; m(iy-2) := m(iy-2) & m(iy+1)
0EC8'  18 E0                jr      logtem1
                   ;
                   ;  -- and object --
                   ;
0ECA'  0EDA' 0EDD' andobj:  defw    andee,andev,andec
0ECE'  0EE2'
0ED0'  0EE5' 0EEA'          defw    andve,andvv,andvc
0ED4'  0EF2'
0ED6'  0EF8' 0EFB'          defw    andce,andcv

0EDA'  02          andee:   defb    2
0EDB'  D1                   pop     de
0EDC'  A2                   and     d

0EDD'  04          andev:   defb    4
0EDE'  21 BBBB              ld      hl,opr2w
0EE1'  A6                   and     (hl)

0EE2'  02          andec:   defb    2
0EE3'  E6 BA                and     opr2b
```

```
0EE5'   04          andve:  defb    4
0EE6'   21 B9B9             ld      hl,opr1w
0EE9'   A6                  and     (hl)
0EEA'   F9          andvv:  defb    -7
0EEB'   3A B9B9             ld      a,(opr1w)
0EEE'   21 BBBB             ld      hl,opr2w
0EF1'   A6                  and     (hl)

0EF2'   FB          andvc:  defb    -5
0EF3'   3A B9B9             ld      a,(opr1w)
0EF6'   E6 BA               and     opr2b

0EF8'   02          andce:  defb    2
0EF9'   E6 B8               and     opr1b

0EFB'   FB          andcv:  defb    -5
0EFC'   3A BBBB             ld      a,(opr2w)
0EFF'   E6 B8               and     opr1b

                    ;
                    ;  logical factor
                    ;
0F01'               logfac::
0F01'   CD 1B0F'            call    gtoken          ;get next token
0F04'               logfac0:
0F04'   3A 0036"            ld      a,(tokcod)
0F07'   FE A9               cp      0a9h            ;'not' ?
0F09'   28 08               jr      z,logfac1
0F0B'   FE 5E               cp      '^'
0F0D'   28 04               jr      z,logfac1
0F0F'   FE 7E               cp      '~'
0F11'   20 27               jr      nz,relexp       ;relational expression
                    ;
                    ;           opr := ^ opr
0F13'               logfac1:
0F13'   CD 1B0F'            call    gtoken          ;get next token
0F16'   CD 0F3A'            call    relexp          ;relational expression
0F19'   FD 7E FD            ld      a,(iy-3)
0F1C'   B7                  or      a
0F1D'   28 11               jr      z,logfac3
0F1F'   3D                  dec     a               ;variable ?
0F20'   28 08               jr      z,logfac2
0F22'   FD 7E FE            ld      a,(iy-2)
0F25'   2F                  cpl
0F26'   FD 77 FE            ld      (iy-2),a        ; m(iy-2) := ^ m(iy-2)
0F29'   C9                  ret
0F2A'               logfac2:
0F2A'   21 0F35'            ld      hl,notobj
0F2D'   C3 188F'            jp      codgen          ;code generate
0F30'               logfac3:
0F30'   3E 2F               ld      a,2fh           ;"  CPL  "
0F32'   C3 1953'            jp      ptcod1
                    ;
                    ;  --  not object  --
                    ;
0F35'   FC          notobj: defb    -4
0F36'   3A B9B9             ld      a,(opr1w)
0F39'   2F                  cpl
                    ;
                    ;  relational expression
                    ;
0F3A'               relexp::
0F3A'   CD 0FBA'            call    ariexp0         ;arithmetic expression
0F3D'   3A 0036"            ld      a,(tokcod)
0F40'   FE 3D               cp      '='
0F42'   28 57               jr      z.relexp50
0F44'   FE F3               cp      0f3h            ;'<>' ?
0F46'   28 42               jr      z.relexp40
0F48'   FE 3E               cp      '>'
```

```
0F4A'   28 2D               jr      z,relexp30
0F4C'   FE F4               cp      0f4h            ;'>=' ?
0F4E'   28 1A               jr      z,relexp20
0F50'   FE 3C               cp      '<'
0F52'   28 0E               jr      z,relexp10
0F54'   FE F2               cp      0f2h            ;'<=' ?
0F56'   C0                  ret     nz
                    ;
                    ;       opr1 := opr1 <= opr2
0F57'               relexp1:
0F57'   CD 0FB7'            call    ariexp          ;arithmetic expression
0F5A'   CD 1863'            call    oprexc          ;exchange
0F5D'   CD 0FFD'            call    subtra
0F60'   18 10               jr      relexp21
                    ;
                    ;       opr1 := opr1 < opr2
0F62'               relexp10:
0F62'   CD 0FB7'            call    ariexp
0F65'   CD 0FFD'            call    subtra
0F68'   18 1A               jr      relexp31
                    ;
                    ;       opr1 := opr1 >= opr2
0F6A'               relexp20:
0F6A'   CD 0FB7'            call    ariexp
0F6D'   CD 0FFD'            call    subtra
0F70'   06                  defb    06h             ;skip 1 byte
0F71'               relobj3:
0F71'   02                  defb    2
0F72'               relexp21:
0F72'   3F                  ccf
0F73'   9F                  sbc     a,a
0F74'   21 0F71'            ld      hl,relobj3
0F77'   18 32               jr      relexp60
                    ;
                    ;       opr1 := opr1 > opr2
0F79'               relexp30:
0F79'   CD 0FB7'            call    ariexp
0F7C'   CD 1863'            call    oprexc          ;exchange
0F7F'   CD 0FFD'            call    subtra
0F82'   06                  defb    06h             ;skip 1 byte
0F83'               relobj2:
0F83'   01                  defb    1
0F84'               relexp31:
0F84'   9F                  sbc     a,a
0F85'   21 0F83'            ld      hl,relobj2
0F88'   18 21               jr      relexp60
                    ;
                    ;       opr1 := opr1 <> opr2
0F8A'               relexp40:
0F8A'   CD 0FB7'            call    ariexp
0F8D'   CD 0FFD'            call    subtra
0F90'   06                  defb    06h             ;skip 1 byte
0F91'               relobj1:
0F91'   04                  defb    4
0F92'   28 02               jr      z,$+4           ;if opr1 <> opr2
0F94'   3E FF                 ld    a,0ffh          ;  then  a := 0ffh  else  a := 0
0F96'   21 0F91'            ld      hl,relobj1
0F99'   18 10               jr      relexp60
                    ;
                    ;       opr1 := opr1 = opr2
0F9B'               relexp50:
0F9B'   CD 0FB7'            call    ariexp
0F9E'   CD 0FFD'            call    subtra
0FA1'   06                  defb    06h             ;skip 1 byte
0FA2'               relobj0:
0FA2'   05                  defb    5
0FA3'   28 02               jr      z,$+4           ;if opr1 = opr2
0FA5'   3E FF                 ld    a,0ffh          ;  then  a := 0ffh
0FA7'   2F                  cpl                     ;  else  a := 0
```

```
0FA8'   21 0FA2'            ld      hl,relobj0
                    ;
0FAB'               relexp60:
0FAB'   47                  ld      b,a
0FAC'   FD 7E FD            ld      a,(iy-3)
0FAF'   B7                  or      a
0FB0'   CA 188F'            jp      z,codgen        ;code generate
0FB3'   FD 70 FE            ld      (iy-2),b
0FB6'   C9                  ret
                    ;
                    ;  arithmetic expression
                    ;
0FB7'               ariexp::
0FB7'   CD 1B0F'            call    gtoken          ;get next token
0FBA'               ariexp0:
0FBA'   CD 1117'            call    term0           ;term
0FBD'               ariexp1:
0FBD'   3A 0036"            ld      a,(tokcod)
0FC0'   FE 2B               cp      '+'
0FC2'   28 4A               jr      z,ariexp5
0FC4'   FE 2D               cp      '-'
0FC6'   28 2D               jr      z,ariexp4
0FC8'   FE AA               cp      0aah            ;'plus' ?
0FCA'   28 16               jr      z,ariexp3
0FCC'   FE AB               cp      0abh            ;'minus' ?
0FCE'   C0                  ret     nz
                    ;
                    ;       opr1 := opr1 minus opr2
0FCF'               ariexp2:
0FCF'   CD 1114'            call    term
0FD2'   21 1024'            ld      hl,minobj
0FD5'   CD 1876'            call    dyagen          ;code generate
0FD8'   30 E3               jr      nc,ariexp1
0FDA'   21 1060'            ld      hl,mincc
0FDD'   CD 189A'            call    codgen0         ;code generate
0FE0'   18 DB               jr      ariexp1
                    ;
                    ;       opr1 := opr1 plus opr2
0FE2'               ariexp3:
0FE2'   CD 1114'            call    term
0FE5'   21 1065'            ld      hl,pluobj
0FE8'   CD 1876'            call    dyagen          ;code generate
0FEB'   30 D0               jr      nc,ariexp1
0FED'   21 109C'            ld      hl,plucc
0FF0'   CD 189A'            call    codgen0         ;code generate
0FF3'   18 C8               jr      ariexp1
                    ;
                    ;       opr1 := opr1 - opr2
0FF5'               ariexp4:
0FF5'   CD 1114'            call    term
0FF8'   CD 0FFD'            call    subtra
0FFB'   18 C0               jr      ariexp1
0FFD'               subtra:
0FFD'   21 10A1'            ld      hl,subobj
1000'   CD 1876'            call    dyagen          ;code generate
1003'   D0                  ret     nc
1004'   FD 7E FE            ld      a,(iy-2)
1007'   FD 96 01            sub     (iy+1)
100A'   FD 77 FE            ld      (iy-2),a        ; m(iy-2) := m(iy-2) - m(iy+1)
100D'   C9                  ret
                    ;
                    ;       opr1 := opr1 + opr2
100E'               ariexp5:
100E'   CD 1114'            call    term
1011'   21 10DD'            ld      hl.addobj
1014'   CD 1876'            call    dyagen          ;code generate
1017'   30 A4               jr      nc,ariexp1
1019'   FD 7E FE            ld      a,(iy-2)
101C'   FD 86 01            add     a,(iy+1)
```

```
101F'   FD 77 FE                ld      (iy-2),a        ; m(iy-2) := m(iy-2) + m(iy+1)
1022'   18 99                   jr      ariexpl
                        ;
                        ;  --  minus object  --
                        ;
1024'   1034' 1038'     minobj: defw    minee,minev,minec
1028'   103D'
102A'   1040' 1046'             defw    minve,minvv,minvc
102E'   104E'
1030'   1054' 1059'             defw    mince,mincv

1034'   03              minee:  defb    3
1035'   57                      ld      d,a
1036'   F1                      pop     af
1037'   9A                      sbc     a,d

1038'   04              minev:  defb    4
1039'   21 BBBB                 ld      hl,opr2w
103C'   9E                      sbc     a,(hl)
103D'   02              minec:  defb    2
103E'   DE BA                   sbc     a,opr2b

1040'   05              minve:  defb    5
1041'   57                      ld      d,a
1042'   3A B9B9                 ld      a,(opr1w)
1045'   9A                      sbc     a,d

1046'   F9              minvv:  defb    -7
1047'   3A B9B9                 ld      a,(opr1w)
104A'   21 BBBB                 ld      hl,opr2w
104D'   9E                      sbc     a,(hl)

104E'   FB              minvc:  defb    -5
104F'   3A B9B9                 ld      a,(opr1w)
1052'   DE BA                   sbc     a,opr2b

1054'   04              mince:  defb    4
1055'   57                      ld      d,a
1056'   3E B8                   ld      a,opr1b
1058'   9A                      sbc     a,d

1059'   FA              mincv:  defb    -6
105A'   3E B8                   ld      a,opr1b
105C'   21 BBBB                 ld      hl,opr2w
105F'   9E                      sbc     a,(hl)

1060'   FC              mincc:  defb    -4
1061'   3E B8                   ld      a,opr1b
1063'   DE BA                   sbc     a,opr2b

                        ;
                        ;  --  plus object  --
                        ;
1065'   1075' 1078'     pluobj: defw    pluee,pluev,pluec
1069'   107D'
106B'   1080' 1085'             defw    pluve,pluvv,pluvc
106F'   108D'
1071'   1093' 1096'             defw    pluce,plucv

1075'   02              pluee:  defb    2
1076'   D1                      pop     de
1077'   8A                      adc     a,d

1078'   04              pluev:  defb    4
1079'   21 BBBB                 ld      hl,opr2w
107C'   8E                      adc     a,(hl)

107D'   02              pluec:  defb    2
```

```
107E'   CE BA                   adc     a,opr2b
1080'   04             pluve:   defb    4
1081'   21 B9B9                 ld      hl,opr1w
1084'   8E                      adc     a,(hl)
1085'   F9             pluvv:   defb    -7
1086'   3A B9B9                 ld      a,(opr1w)
1089'   21 BBBB                 ld      hl,opr2w
108C'   8E                      adc     a,(hl)

108D'   FB             pluvc:   defb    -5
108E'   3A B9B9                 ld      a,(opr1w)
1091'   CE BA                   adc     a,opr2b

1093'   02             pluce:   defb    2
1094'   CE BA                   adc     a,opr2b

1096'   FB             plucv:   defb    -5
1097'   3A BBBB                 ld      a,(opr2w)
109A'   CE B8                   adc     a,opr1b

109C'   FC             plucc:   defb    -4
109D'   3E B8                   ld      a,opr1b
109F'   CE BA                   adc     a,opr2b
                       ;
                       ;  --  subtract object  --
                       ;
10A1'   10B1' 10B5'    subobj:  defw    subee,subev,subec
10A5'   10BA'
10A7'   10BD' 10C3'             defw    subve,subvv,subvc
10AB'   10CB'
10AD'   10D1' 10D6'             defw    subce,subcv

10B1'   03             subee:   defb    3
10B2'   57                      ld      d,a
10B3'   F1                      pop     af
10B4'   92                      sub     d

10B5'   04             subev:   defb    4
10B6'   21 BBBB                 ld      hl,opr2w
10B9'   96                      sub     (hl)

10BA'   02             subec:   defb    2
10BB'   D6 BA                   sub     opr2b

10BD'   05             subve:   defb    5
10BE'   57                      ld      d,a
10BF'   3A B9B9                 ld      a,(opr1w)
10C2'   92                      sub     d

10C3'   F9             subvv:   defb    -7
10C4'   3A B9B9                 ld      a,(opr1w)
10C7'   21 BBBB                 ld      hl,opr2w
10CA'   96                      sub     (hl)

10CB'   FB             subvc:   defb    -5
10CC'   3A B9B9                 ld      a,(opr1w)
10CF'   D6 BA                   sub     opr2b

10D1'   04             subce:   defb    4
10D2'   57                      ld      d,a
10D3'   3E B8                   ld      a,opr1b
10D5'   92                      sub     d

10D6'   FA             subcv:   defb    -6
10D7'   3E B8                   ld      a,opr1b
10D9'   21 BBBB                 ld      hl,opr2w
10DC'   96                      sub     (hl)

                       ;
```

```
                          ; --  add object  --
                          ;
10DD'  10ED' 10F0'        addobj: defw   addee,addev,addec
10E1'  10F5'
10E3'  10F8' 10FD'                defw   addve,addvv,addvc
10E7'  1105'
10E9'  110B' 110E'                defw   addce,addcv

10ED'  02                 addee:  defb   2
10EE'  D1                         pop    de
10EF'  82                         add    a,d

10F0'  04                 addev:  defb   4
10F1'  21 BBBB                    ld     hl,opr2w
10F4'  86                         add    a,(hl)

10F5'  02                 addec:  defb   2
10F6'  C6 BA                      add    a,opr2b

10F8'  04                 addve:  defb   4
10F9'  21 B9B9                    ld     hl,opr1w
10FC'  86                         add    a,(hl)

10FD'  F9                 addvv:  defb   -7
10FE'  3A B9B9                    ld     a,(opr1w)
1101'  21 BBBB                    ld     hl,opr2w
1104'  86                         add    a,(hl)

1105'  FB                 addvc:  defb   -5
1106'  3A B9B9                    ld     a,(opr1w)
1109'  C6 BA                      add    a,opr2b

110B'  02                 addce:  defb   2
110C'  C6 B8                      add    a,opr1b

110E'  FB                 addcv:  defb   -5
110F'  3A BBBB                    ld     a,(opr2w)
1112'  C6 B8                      add    a,opr1b

                          ;
                          ;  term
                          ;
1114'                     term::
1114'  CD 1B0F'                   call   gtoken          ;get next token
1117'                     term0:
1117'  CD 1335'                   call   factor0         ;factor
111A'                     term1:
111A'  3A 0036"                   ld     a,(tokcod)
111D'  FE 2A                      cp     '*'
111F'  28 73                      jr     z,term6
1121'  FE 2F                      cp     '/'
1123'  28 56                      jr     z,term5
1125'  FE 25                      cp     '%'
1127'  28 39                      jr     z,term4
1129'  FE F1                      cp     0f1h            ;'<<' ?
112B'  28 1C                      jr     z,term3
112D'  FE F5                      cp     0f5h            ;'>>' ?
112F'  C0                         ret    nz
                          ;
                          ;        opr1 := opr1 >> opr2
1130'                     term2:
1130'  CD 1332'                   call   factor
1133'  21 11AF'                   ld     hl,shrobj
1136'  CD 1876'                   call   dyagen          ;code generate
1139'  30 DF                      jr     nc,term1
113B'  FD 7E FE                   ld     a,(iy-2)
113E'  FD 56 01                   ld     d,(iy+1)
1141'  CD 0000*                   call   @shr.r
1144'  FD 77 FE                   ld     (iy-2),a        ; m(iy-2) := m(iy-2) >> m(iy+1)
```

```
1147'   18 D1                  jr      term1
                        ;
                        ;       opr1 :=opr1 << opr2
1149'                   term3:
1149'   CD 1332'               call    factor
114C'   21 11FD'               ld      hl,shlobj
114F'   CD 1876'               call    dyagen          ;code generate
1152'   30 C6                  jr      nc,term1
1154'   FD 7E FE               ld      a,(iy-2)
1157'   FD 56 01               ld      d,(iy+1)
115A'   CD 0000*               call    @shl.r
115D'   FD 77 FE               ld      (iy-2),a        ; m(iy-2) := m(iy-2) << m(iy+1)
1160'   18 B8                  jr      term1
                        ;
                        ;       opr1 := opr1 % opr2
1162'                   term4:
1162'   CD 1332'               call    factor
1165'   21 124B'               ld      hl,remobj
1168'   CD 1876'               call    dyagen          ;code generate
116B'   30 AD                  jr      nc,term1
116D'   FD 7E FE               ld      a,(iy-2)
1170'   FD 56 01               ld      d,(iy+1)
1173'   CD 0000*               call    @rem.r
1176'   FD 77 FE               ld      (iy-2),a        ; m(iy-2) := m(iy-2) % m(iy+1)
1179'   18 9F                  jr      term1
                        ;
                        ;       opr1 := opr1 / opr2
117B'                   term5:
117B'   CD 1332'               call    factor
117E'   21 1299'               ld      hl,divobj
1181'   CD 1876'               call    dyagen          ;code generate
1184'   30 94                  jr      nc,term1
1186'   FD 7E FE               ld      a,(iy-2)
1189'   FD 56 01               ld      d,(iy+1)
118C'   CD 0000*               call    @div.r
118F'   FD 77 FE               ld      (iy-2),a        ; m(iy-2) := m(iy-2) / m(iy+1)
1192'   18 86                  jr      term1
                        ;
                        ;       opr1 := opr1 * opr2
1194'                   term6:
1194'   CD 1332'               call    factor
1197'   21 12E7'               ld      hl,mulobj
119A'   CD 1876'               call    dyagen          ;code generate
119D'   D2 111A'               jp      nc,term1
11A0'   FD 7E FE               ld      a,(iy-2)
11A3'   FD 56 01               ld      d,(iy+1)
11A6'   CD 0000*               call    @mul.r
11A9'   FD 77 FE               ld      (iy-2),a        ; m(iy-2) := m(iy-2) * m(iy+1)
11AC'   C3 111A'               jp      term1
                        ;
                        ;  --  shift right ojbect  --
                        ;
11AF'   11BF' 11C5'     shrobj: defw    shree,shrev,shrec
11B3'   11CC'
11B5'   11D2' 11DA'            defw    shrve,shrvv,shrvc
11B9'   11E4'
11BB'   11ED' 11F4'            defw    shrce,shrcv

11BF'   05              shree:  defb    5
11C0'   57                     ld      d,a
11C1'   F1                     pop     af
11C2'   CD 0000*               call    @shr.r-@start

11C5'   06              shrev:  defb    6
11C6'   21 BBBB                ld      hl,opr2w
11C9'   CD 0000*               call    @shr.m-@start

11CC'   05              shrec:  defb    5
```

```
11CD'   16 BA                  ld      d,opr2b
11CF'   CD 0000*               call    @shr.r-@start

11D2'   07            shrve:   defb    7
11D3'   57                     ld      d,a
11D4'   3A B9B9                ld      a,(opr1w)
11D7'   CD 0000*               call    @shr.r-@start

11DA'   F7            shrvv:   defb    -9
11DB'   3A B9B9                ld      a,(opr1w)
11DE'   21 BBBB                ld      hl,opr2w
11E1'   CD 0000*               call    @shr.m-@start

11E4'   F8            shrvc:   defb    -8
11E5'   3A B9B9                ld      a,(opr1w)
11E8'   16 BA                  ld      d,opr2b
11EA'   CD 0000*               call    @shr.r-@start

11ED'   06            shrce:   defb    6
11EE'   57                     ld      d,a
11EF'   3E B8                  ld      a,opr1b
11F1'   CD 0000*               call    @shr.r-@start

11F4'   F8            shrcv:   defb    -8
11F5'   3E B8                  ld      a,opr1b
11F7'   21 BBBB                ld      hl,opr2w
11FA'   CD 0000*               call    @shr.m-@start

                      ;
                      ;  --  shift left ojbect  --
                      ;
11FD'   120D' 1213'   shlobj:  defw    shlee,shlev,shlec
1201'   121A'
1203'   1220' 1228'            defw    shlve,shlvv,shlvc
1207'   1232'
1209'   123B' 1242'            defw    shlce,shlcv

120D'   05            shlee:   defb    5
120E'   57                     ld      d,a
120F'   F1                     pop     af
1210'   CD 0000*               call    @shl.r-@start

1213'   06            shlev:   defb    6
1214'   21 BBBB                ld      hl,opr2w
1217'   CD 0000*               call    @shl.m-@start

121A'   05            shlec:   defb    5
121B'   16 BA                  ld      d,opr2b
121D'   CD 0000*               call    @shl.r-@start

1220'   07            shlve:   defb    7
1221'   57                     ld      d,a
1222'   3A B9B9                ld      a,(opr1w)
1225'   CD 0000*               call    @shl.r-@start

1228'   F7            shlvv:   defb    -9
1229'   3A B9B9                ld      a,(opr1w)
122C'   21 BBBB                ld      hl,opr2w
122F'   CD 0000*               call    @shl.m-@start

1232'   F8            shlvc:   defb    -8
1233'   3A B9B9                ld      a,(opr1w)
1236'   16 BA                  ld      d,opr2b
1238'   CD 0000*               call    @shl.r-@start

123B'   06            shlce:   defb    6
123C'   57                     ld      d,a
123D'   3E B8                  ld      a,opr1b
```

```
123F'   CD 0000*                call    @shl.r-@start

1242'   F8              shlcv:  defb    -8
1243'   3E B8                   ld      a,opr1b
1245'   21 BBBB                 ld      hl,opr2w
1248'   CD 0000*                call    @shl.m-@start

                        ;
                        ;  --  remainder ojbect  --
                        ;
124B'   125B' 1261'     remobj: defw    remee,remev.remec
124F'   1268'
1251'   126E' 1276'             defw    remve,remvv.remvc
1255'   1280'
1257'   1289' 1290'             defw    remce.remcv

125B'   05              remee:  defb    5
125C'   57                      ld      d.a
125D'   F1                      pop     af
125E'   CD 0000*                call    @rem.r-@start

1261'   06              remev:  defb    6
1262'   21 BBBB                 ld      hl.opr2w
1265'   CD 0000*                call    @rem.m-@start

1268'   05              remec:  defb    5
1269'   16 BA                   ld      d.opr2b
126B'   CD 0000*                call    @rem.r-@start

126E'   07              remve:  defb    7
126F'   57                      ld      d.a
1270'   3A B9B9                 ld      a.(opr1w)
1273'   CD 0000*                call    @rem.r-@start

1276'   F7              remvv:  defb    -9
1277'   3A B9B9                 ld      a,(opr1w)
127A'   21 BBBB                 ld      hl,opr2w
127D'   CD 0000*                call    @rem.m-@start

1280'   F8              remvc:  defb    -8
1281'   3A B9B9                 ld      a.(opr1w)
1284'   16 BA                   ld      d.opr2b
1286'   CD 0000*                call    @rem.r-@start

1289'   06              remce:  defb    6
128A'   57                      ld      d,a
128B'   3E B8                   ld      a,opr1b
128D'   CD 0000*                call    @rem.r-@start

1290'   F8              remcv:  defb    -8
1291'   3E B8                   ld      a,opr1b
1293'   21 BBBB                 ld      hl,opr2w
1296'   CD 0000*                call    @rem.m-@start

                        ;
                        ;  --  divide ojbect  --
                        ;
1299'   12A9' 12AF'     divobj: defw    divee,divev.divec
129D'   12B6'
129F'   12BC' 12C4'             defw    divve,divvv.divvc
12A3'   12CE'
12A5'   12D7' 12DE'             defw    divce.divcv

12A9'   05              divee:  defb    5
12AA'   57                      ld      d,a
12AB'   F1                      pop     af
12AC'   CD 0000*                call    @div.r-@start
```

```
12AF'   06              divev:  defb    6
12B0'   21 BBBB                 ld      hl,opr2w
12B3'   CD 0000*                call    @div.m-@start

12B6'   05              divec:  defb    5
12B7'   16 BA                   ld      d,opr2b
12B9'   CD 0000*                call    @div.r-@start

12BC'   07              divve:  defb    7
12BD'   57                      ld      d,a
12BE'   3A B9B9                 ld      a,(opr1w)
12C1'   CD 0000*                call    @div.r-@start

12C4'   F7              divvv:  defb    -9
12C5'   3A B9B9                 ld      a,(opr1w)
12C8'   21 BBBB                 ld      hl,opr2w
12CB'   CD 0000*                call    @div.m-@start

12CE'   F8              divvc:  defb    -8
12CF'   3A B9B9                 ld      a,(opr1w)
12D2'   16 BA                   ld      d,opr2b
12D4'   CD 0000*                call    @div.r-@start

12D7'   06              divce:  defb    6
12D8'   57                      ld      d,a
12D9'   3E B8                   ld      a,opr1b
12DB'   CD 0000*                call    @div.r-@start

12DE'   F8              divcv:  defb    -8
12DF'   3E B8                   ld      a,opr1b
12E1'   21 BBBB                 ld      hl,opr2w
12E4'   CD 0000*                call    @div.m-@start

                        ;
                        ;  --  multiply ojbect  --
                        ;
12E7'   12F7' 12FC'     mulobj: defw    mulee,mulev.mulec
12EB'   1303'
12ED'   1309' 1310'             defw    mulve.mulvv.mulvc
12F1'   131A'
12F3'   1323' 1329'             defw    mulce.mulcv

12F7'   04              mulee:  defb    4
12F8'   D1                      pop     de
12F9'   CD 0000*                call    @mul.r-@start

12FC'   06              mulev:  defb    6
12FD'   21 BBBB                 ld      hl.opr2w
1300'   CD 0000*                call    @mul.m-@start

1303'   05              mulec:  defb    5
1304'   16 BA                   ld      d,opr2b
1306'   CD 0000*                call    @mul.r-@start
1309'   06              mulve:  defb    6
130A'   21 B9B9                 ld      hl,opr1w
130D'   CD 0000*                call    @mul.m-@start

1310'   F7              mulvv:  defb    -9
1311'   3A B9B9                 ld      a,(opr1w)
1314'   21 BBBB                 ld      hl,opr2w
1317'   CD 0000*                call    @mul.m-@start

131A'   F8              mulvc:  defb    -8
131B'   3A B9B9                 ld      a,(opr1w)
131E'   16 BA                   ld      d,opr2b
1320'   CD 0000*                call    @mul.r-@start

1323'   05              mulce:  defb    5
```

```
1324'   16 B8                   ld      d,opr1b
1326'   CD 0000*                call    @mul.r-@start

1329'   F8             mulcv:   defb    -8
132A'   3A BBBB                 ld      a,(opr2w)
132D'   16 B8                   ld      d,opr1b
132F'   CD 0000*                call    @mul.r-@start

                       ;
                       ;  factor
                       ;
1332'                  factor::
1332'   CD 1B0F'                call    gtoken          ;get next token
1335'                  factor0:
1335'   3A 0036"                ld      a,(tokcod)
1338'   FE 2B                   cp      '+'
133A'   28 2B                   jr      z,factor5
133C'   FE 2D                   cp      '-'
133E'   20 2A                   jr      nz,expr
1340'   CD 1367'                call    factor5
1343'   FD 7E FD                ld      a,(iy-3)
1346'   B7                      or      a               ;expression ?
1347'   28 18                   jr      z,factor2
1349'   3D                      dec     a               ;variable ?
134A'   28 09                   jr      z,factor1
134C'   FD 7E FE                ld      a,(iy-2)
134F'   ED 44                   neg
1351'   FD 77 FE                ld      (iy-2),a        ; m(iy-2) := -m(iy-2)
1354'   C9                      ret
1355'                  factor1:
1355'   21 135B'                ld      hl,$+6
1358'   C3 188F'                jp      codgen
                                ;----------------------+
135B'   FB                      defb    -5             ;
135C'   3A B9B9                 ld      a,(opr1w)      ; neg object
135F'   ED 44                   neg                    ;
                                ;----------------------+
1361'                  factor2:
1361'   11 ED44                 ld      de,0ed44h       ;"  NEG  "
1364'   C3 1964'                jp      ptcod2
                       ;
1367'                  factor5:
1367'   CD 1B0F'                call    gtoken          ;get next token
                       ;
                       ;          (  expression  )
136A'                  expr::
136A'   FE 28                   cp      '('
136C'   20 26                   jr      nz,fun?
136E'   CD 1B0F'                call    gtoken          ;get next token
1371'                  exprx:
1371'   CD 0DF5'                call    logexp          ;expression
1374'                  exprx1:
1374'   FE 2C                   cp      ','
1376'   20 14                   jr      nz,expry
1378'   CD 0DD2'                call    expr5
137B'   CD 1856'                call    decopt          ; iy := iy - 3 ; dec(optble)
137E'   CD 1B0F'                call    gtoken
1381'   CD 0DCB'                call    expr1           ;expression
1384'   3A 0036"                ld      a,(tokcod)
1387'   18 EB                   jr      exprx1
1389'                  chkpar:
1389'   3A 0036"                ld      a,(tokcod)
138C'                  expry:
138C'   FE 29                   cp      ')'
138E'   C2 203D'                jp      nz.experr
1391'   C3 1B0F'                jp      gtoken
                       ;
                       ;          ?( exp_1; exp_2, exp_3 )
```

```
1394'                   fun?::
1394'   FE 3F                   cp      '?'
1396'   20 53                   jr      nz,memor
                        ;
1398'   CD 1B0F'                call    gtoken          ;get next token
139B'   FE 28                   cp      '('
139D'   C2 203D'                jp      nz,experr
13A0'   CD 1B0F'                call    gtoken
13A3'   CD 0DCF'                call    expr2           ;expression_1
13A6'   3A 0036"                ld      a,(tokcod)
13A9'   FE 3B                   cp      ';'
13AB'   C2 203D'                jp      nz,experr
13AE'   3E B7                   ld      a,0b7h          ;"  OR   A  "
13B0'   CD 1953'                call    ptcod1
13B3'   23                      inc     hl
13B4'   E5                      push    hl
13B5'   3E CA                   ld      a,0cah
13B7'   21 0000                 ld      hl,0            ;"L1: JP   Z,0  "
13BA'   CD 196E'                call    ptcod3
13BD'   CD 1B0F'                call    gtoken
13C0'   CD 0DCB'                call    expr1           ;expression_2
13C3'   CD 1856'                call    decopt          ; iy := iy - 3 ; dec(optble)
13C6'   3A 0036"                ld      a,(tokcod)
13C9'   FE 2C                   cp      ','
13CB'   C2 203D'                jp      nz,experr
13CE'   3E C3                   ld      a,0c3h
13D0'   21 0000                 ld      hl,0            ;"L2: JP   0  "
13D3'   CD 196E'                call    ptcod3
13D6'   2B                      dec     hl
13D7'   2B                      dec     hl
13D8'   E3                      ex      (sp),hl
13D9'   CD 19B7'                call    ptcha           ;"  chain  L1+1  "
13DC'   CD 1B0F'                call    gtoken
13DF'   CD 0DCB'                call    expr1           ;expression_3
13E2'   CD 1856'                call    decopt          ; iy := iy - 3 ; dec(optble)
13E5'   E1                      pop     hl
13E6'   CD 19B7'                call    ptcha           ;"  chain  L2+1  "
13E9'   18 9E                   jr      chkpar
                        ;
                        ;       memory[ exp_1, exp_2 ] := exp_3
13EB'                   memor::
13EB'   FE B9                   cp      0b9h            ;'memory' ?
13ED'   28 05                   jr      z,$+7
13EF'   FE 40                     cp    '@'
13F1'   C2 14F3'                  jp    nz,port
                        ;
13F4'   CD 1B0F'                call    gtoken
13F7'   FE 5B                   cp      '['
13F9'   C2 203D'                jp      nz,experr
13FC'   CD 1B0F'                call    gtoken
13FF'   FE BF                   cp      0bfh            ;'ix' ?
1401'   28 4B                   jr      z,memor2
1403'   FE C0                   cp      0c0h            ;'iy' ?
1405'   28 4A                   jr      z,memor3
                        ;
1407'   CD 0DCF'                call    expr2           ;expression_1
140A'   CD 1856'                call    decopt          ; iy := iy - 3 ; dec(optble)
140D'   3A 0036"                ld      a,(tokcod)
1410'   FE 2C                   cp      ','
1412'   C2 203D'                jp      nz,experr
1415'   CD 1B0F'                call    gtoken
1418'   3E F5                   ld      a,0f5h          ;"  PUSH AF  "
141A'   CD 1953'                call    ptcod1
141D'   CD 0DCB'                call    expr1           ;expression_2
1420'   3A 0036"                ld      a,(tokcod)
1423'   FE 5D                   cp      ']'
1425'   C2 203D'                jp      nz,experr
1428'   11 E16F                 ld      de,0e16fh       ;"  POP  HL  "
142B'   CD 1964'                call    ptcod2          ;"  LD   L,A  "
```

```
 142E'   CD 1B0F'            call    gtoken
 1431'   FE F0               cp      0f0h            ;':='?
 1433'   20 14               jr      nz,memor1
 1435'   CD 1856'            call    decopt          ; iy := iy - 3 ; dec(optble)
 1438'   3E E5               ld      a,0e5h          ;"  PUSH HL  "
 143A'   CD 1953'            call    ptcod1
 143D'   CD 1B0F'            call    gtoken
 1440'   CD 0DCB'            call    expr1           ;expression_3
 1443'   11 E177             ld      de,0e177h       ;"  POP  HL  "
 1446'   C3 1964'            jp      ptcod2          ;"  LD   (HL),A  "
                     ;
                     ;          memory[ exp_1, exp_2 ]
 1449'               memor1:
 1449'   3E 7E               ld      a,7eh           ;"  LD   A,(HL)  "
 144B'   C3 1953'            jp      ptcod1
                     ;
                     ;          memory[ index +-, const ] := exp_1
 144E'               memor2:
 144E'   3E DD               ld      a,0ddh          ;index register ix
 1450'   21                  defb    21h
 1451'               memor3:
 1451'   3E FD               ld      a,0fdh          ;index register iy
 1453'   F5                  push    af
 1454'   CD 1B0F'            call    gtoken
 1457'   FE 2B               cp      '+'             ;auto increment ?
 1459'   28 16               jr      z,memor5
 145B'   FE 2D               cp      '-'             ;auto decremnnt ?
 145D'   28 0F               jr      z,memor4
 145F'   2E 00               ld      l,0
 1461'   FE 2C               cp      ','
 1463'   28 1E               jr      z,memor6
 1465'   26 00               ld      h,0
 1467'   FE 5D               cp      ']'
 1469'   28 4C               jr      z,memor10
 146B'   C3 203D'            jp      experr
 146E'               memor4:
 146E'   2E 2B               ld      l,2bh           ;decrement
 1470'   11                  defb    11h             ; 2 byte skip
 1471'               memor5:
 1471'   2E 23               ld      l,23h           ;incremnet
 1473'   E5                  push    hl
 1474'   CD 1B0F'            call    gtoken
 1477'   E1                  pop     hl
 1478'   26 00               ld      h,0
 147A'   FE 5D               cp      ']'
 147C'   28 39               jr      z,memor10
 147E'   FE 2C               cp      ','
 1480'   C2 203D'            jp      nz,experr
 1483'               memor6:
 1483'   E5                  push    hl
 1484'   CD 1B0F'            call    gtoken
 1487'   FE 2B               cp      '+'
 1489'   28 05               jr      z,memor7
 148B'   FE 2D               cp      '-'
 148D'   20 08               jr      nz,memor8
 148F'   FE                  defb    0feh            ;reset z flag & 1 byte skip
 1490'               memor7:
 1490'   AF                  xor     a               ;set z flag
 1491'   F5                  push    af
 1492'   CD 1B0F'            call    gtoken
 1495'   F1                  pop     af
 1496'   0E                  defb    0eh             ; 1 byte skip
 1497'               memor8:
 1497'   AF                  xor     a               ;set z flag
 1498'   F5                  push    af
 1499'   CD 19ED'            call    bytcon          ;byte constant
 149C'   F1                  pop     af
 149D'   28 04               jr      z,memor9
```

```
149F'   AF                      xor     a
14A0'   95                      sub     l
14A1'   6F                      ld      l,a             ; l := 0 - l
14A2'   9F                      sbc     a,a
14A3'                   memor9:
14A3'   CB 7D                   bit     7,l             ;check  l = -128...+127 ?
14A5'   28 01                   jr      z,$+3
14A7'   2F                        cpl
14A8'   B7                      or      a
14A9'   C4 1F99'                call    nz,ilcnds
14AC'   D1                      pop     de
14AD'   55                      ld      d,l
14AE'   EB                      ex      de,hl
14AF'   3A 0036"                ld      a,(tokcod)
14B2'   FE 5D                   cp      ']'
14B4'   C2 203D'                jp      nz,experr
14B7'                   memor10:
14B7'   E5                      push    hl
14B8'   CD 1B0F'                call    gtoken
14BB'   FE F0                   cp      0f0h            ;':=' ?
14BD'   20 19                   jr      nz,memor11
14BF'   CD 1B0F'                call    gtoken
14C2'   CD 0DCF'                call    expr2           ;expression_1
14C5'   E1                      pop     hl
14C6'   D1                      pop     de
14C7'   7A                      ld      a,d
14C8'   D5                      push    de
14C9'   E5                      push    hl
14CA'   2E 77                   ld      l,77h           ;"  LD (IX+d),A " or "  LD (IY+d),A "
14CC'   CD 196E'                call    ptcod3
14CF'   E1                      pop     hl
14D0'   D1                      pop     de
14D1'   7D                      ld      a,l
14D2'   B7                      or      a
14D3'   C8                      ret     z
14D4'   5D                      ld      e,l             ;"  INC index " or "  DEC index "
14D5'   C3 1964'                jp      ptcod2
                        ;
                        ;               memory[ index +- , const ]
14D8'                   memor11:
14D8'   CD 1944'                call    pacod           ;"  PUSH AF  "
14DB'   E1                      pop     hl
14DC'   D1                      pop     de
14DD'   7A                      ld      a,d
14DE'   D5                      push    de
14DF'   E5                      push    hl
14E0'   2E 7E                   ld      l,7eh           ;"  LD A,(IX+d) " or "  LD  A,(IY+d) "
14E2'   CD 196E'                call    ptcod3
14E5'   E1                      pop     hl
14E6'   D1                      pop     de
14E7'   7D                      ld      a,l
14E8'   B7                      or      a
14E9'   28 05                   jr      z,memor12
14EB'   5D                      ld      e,l             ;"  INC index " or "  DEC index "
14EC'   CD 1964'                call    ptcod2
14EF'   AF                      xor     a
14F0'                   memor12:
14F0'   C3 183D'                jp      stoptb
                        ;
                        ;               port[ exp_1 ] := exp_2
14F3'                   port::
14F3'   FE BA                   cp      0bah            ;'port' ?
14F5'   28 04                   jr      z,$+6
14F7'   FE F6                     cp    0f6h            ;'@@' ?
14F9'   20 3F                     jr    nz,itrfun
                        ;
14FB'   CD 1B0F'                call    gtoken
14FE'   FE 5B                   cp      '['
```

```
1500'   C2 203D'               jp      nz,experr
1503'   CD 1B0F'               call    gtoken
1506'   CD 0DCF'               call    expr2          ;expression_1
1509'   3A 0036"               ld      a,(tokcod)
150C'   FE 5D                  cp      ']'
150E'   C2 203D'               jp      nz,experr
1511'   CD 1B0F'               call    gtoken
1514'   FE F0                  cp      0f0h           ;':=' ?
1516'   20 1A                  jr      nz,port1
1518'   CD 1856'               call    decopt         ; iy := iy - 3 ; dec(optble)
151B'   3E F5                  ld      a,0f5h         ;"  PUSH AF  "
151D'   CD 1953'               call    ptcod1
1520'   CD 1B0F'               call    gtoken
1523'   CD 0DCB'               call    expr1          ;expression_2
1526'   11 D14A                ld      de,0d14ah      ;"  POP  DE  "
1529'   CD 1964'               call    ptcod2         ;"  LD   C,D "
152C'   11 ED79                ld      de,0ed79h      ;"  OUT  (C),A"
152F'   C3 1964'               jp      ptcod2
                       ;
                       ;           port[ exp_1 ]
1532'                  port1:
1532'   3E 4F                  ld      a,4fh          ;"  LD   C,A  "
1534'   21 78ED                ld      hl,78edh       ;"  IN   A,(C)"
1537'   C3 196E'               jp      ptcod3
                       ;
                       ;   intrinsic function    inc( var ), dec( var )
153A'                  itrfun::
153A'   FE AC                  cp      0ach           ;'inc' ?
153C'   28 07                  jr      z,itrfun1
153E'   FE AD                  cp      0adh           ;'dec' ?
1540'   20 48                  jr      nz,itrfun2
1542'   3E 01                  ld      a,1
1544'   FE                     defb    0feh
1545'                  itrfun1:
1545'   AF                     xor     a
1546'   F5                     push    af
1547'   CD 1B0F'               call    gtoken         ;get next token
154A'   FE 28                  cp      '('
154C'   C2 203D'               jp      nz,experr
154F'   CD 1B0F'               call    gtoken
1552'   3D                     dec     a
1553'   C2 203D'               jp      nz,experr
1556'   21 0016"               ld      hl,idetyp
1559'   CD 0000*               call    seasym         ;search symbol table
155C'   B7                     or      a
155D'   C4 1E74'               call    nz.misvar
1560'   3A 0016"               ld      a,(idetyp)
1563'   E6 0F                  and     0fh
1565'   FE 02                  cp      2              ;variable name ?
1567'   C2 203D'               jp      nz,experr
156A'   2A 0023"               ld      hl,(adrs)
156D'   3E 21                  ld      a,21h          ;"  LD   HL,nn  "
156F'   CD 196E'               call    ptcod3
1572'   F1                     pop     af
1573'   C6 34                  add     a,34h          ;"  INC (HL) " or "  DEC (HL) "
1575'   CD 1953'               call    ptcod1
1578'   AF                     xor     a
1579'   CD 183D'               call    stoptb
157C'   21 1588'               ld      hl,ldam        ;"  LD   A,(HL)  "
157F'   CD 188F'               call    codgen
1582'   CD 1B0F'               call    gtoken
1585'   C3 1389'               jp      chkpar         ;')' ?
                       ;
1588'   FF             ldam:   defb    -1
1589'   7E                     ld      a,(hl)
                       ;
                       ;   intrinsic function    rl( exp ), rlc( exp ), rr( exp )
                       ;                         rrc( exp ), sra( exp ), decj( exp )
158A'                  itrfun2:
```

```
158A'  FE AE              cp      0aeh          ;'rl'.'rlc'.'rr'.'rrc'.'sra'.'decj' ?
158C'  DA 161E'           jp      c,vardat
158F'  FE B4              cp      0b3h+1
1591'  30 2E              jr      nc,itrfun3
1593'  D6 AE              sub     0aeh
1595'  F5                 push    af
1596'  CD 1B0F'           call    gtoken
1599'  FE 28              cp      '('
159B'  C2 203D'           jp      nz,experr
159E'  CD 1B0F'           call    gtoken
15A1'  CD 1371'           call    exprx         ;expression
15A4'  CD 0DD2'           call    expr5
15A7'  F1                 pop     af
15A8'  5F                 ld      e,a
15A9'  16 00              ld      d,0
15AB'  21 15BB'           ld      hl,iteobj
15AE'  19                 add     hl,de
15AF'  7E                 ld      a,(hl)
15B0'  FE CB              cp      0cbh
15B2'  C2 1953'           jp      nz,ptcod1
15B5'  57                 ld      d,a
15B6'  1E 2F              ld      e,2fh         ;" SRA  A  "
15B8'  C3 1964'           jp      ptcod2
                   ;
15BB'  17          iteobj: rla
15BC'  07                 rlca
15BD'  1F                 rra
15BE'  0F                 rrca
15BF'  CB                 defb    0cbh
15C0'  27                 daa
                   ;
                   ;    intrinsic function    carry( exp ), zero( exp )
                   ;                          sign( exp ), parity( exp ), overflow( exp )
15C1'              itrfun3:
15C1'  FE B9              cp      0b8h+1        ;'carry'.'zero'.'sign'.'parity'.'overflow' ?
15C3'  30 59              jr      nc,vardat
15C5'  D6 B4              sub     0b4h
15C7'  F5                 push    af
15C8'  CD 1B0F'           call    gtoken
15CB'  FE 28              cp      '('
15CD'  C2 203D'           jp      nz,experr
15D0'  CD 1B0F'           call    gtoken
15D3'  FE 29              cp      ')'
15D5'  28 08              jr      z,itrfun4
15D7'  CD 1371'           call    exprx         ;expression
15DA'  CD 0DD2'           call    expr5
15DD'  18 0A              jr      itrfun8
15DF'              itrfun4:
15DF'  CD 1B0F'           call    gtoken
15E2'  CD 1944'           call    pacod         ;" PUSH AF  "
15E5'  AF                 xor     a
15E6'  CD 183D'           call    stoptb
15E9'              itrfun8:
15E9'  F1                 pop     af
15EA'  20 04              jr      nz,itrfun5
15EC'  3E 9F              ld      a,9fh         ;" SBC  A,A  "
15EE'  18 13              jr      itrfun6
15F0'              itrfun5:
15F0'  F5                 push    af
15F1'  11 3E00            ld      de,3e00h      ;" LD   A,0  "
15F4'  CD 1964'           call    ptcod2
15F7'  F1                 pop     af
15F8'  3D                 dec     a
15F9'  20 0B              jr      nz,itrfun7
15FB'  11 2001            ld      de,2001h      ;" JR   NZ,$+3 "
15FE'  CD 1964'           call    ptcod2
1601'  3E 2F              ld      a,2fh         ;" CPL  "
1603'              itrfun6:
```

```
1603'  C3 1953'               jp      ptcod1
1606'                 itrfun7:
1606'  21 0000                ld      hl,0
1609'  3D                     dec     a
160A'  3E F2                  ld      a,0f2h        ;"L1: JP   P,0  "
160C'  28 02                  jr      z,$+4
160E'  3E E2                    ld    a,0e2h        ;"L1: JP   PO,0 "
1610'  CD 196E'               call    ptcod3
1613'  3E 2F                  ld      a,2fh         ;"  CPL  "
1615'  CD 1953'               call    ptcod1
1618'  2B                     dec     hl
1619'  2B                     dec     hl
161A'  2B                     dec     hl
161B'  C3 19B7'               jp      ptcha         ;"  chain L1+1  "
                      ;
                      ;       variable & data name
161E'                 vardat::
161E'  2A 0012"               ld      hl,(conval)
1621'  B7                     or      a
1622'  CA 16F5'               jp      z,const5
1625'  3D                     dec     a             ;identifier ?
1626'  C2 16F0'               jp      nz,const
1629'  21 0016"               ld      hl,idetyp
162C'  CD 0000*               call    seasym        ;search symbol table
162F'  B7                     or      a             ;found ?
1630'  20 19                  jr      nz,uvarck     ;skip if not
1632'  2A 0023"               ld      hl,(adrs)
1635'  3A 0016"               ld      a,(idetyp)
1638'  E6 0F                  and     0fh
163A'  FE 01                  cp      1             ;constant name ?
163C'  CA 16F5'               jp      z,const5      ;skip if so
163F'  FE 08                  cp      8             ;label ?
1641'  CA 203D'               jp      z,experr      ;error if so
1644'  FE 09                  cp      9             ;function name ?
1646'  CA 172D'               jp      z,funcal      ;skip if so
1649'  18 13                  jr      vardat0
164B'                 uvarck:
164B'  2A 0000"               ld      hl,(cptr)
164E'  CD 1DBE'               call    spskip        ;space skip
1651'  FE 28                  cp      '('
1653'  CA 1704'               jp      z,ufuncl
1656'  CD 1E74'               call    misvar
1659'  3E 02                  ld      a,2
165B'  21 0000                ld      hl,0
                      ;
                      ;           var[ exp_1 ] := exp_2
165E'                 vardat0:
165E'  F5                     push    af
165F'  E5                     push    hl
1660'  CD 1B0F'               call    gtoken        ;get next token
1663'  FE 5B                  cp      '['
1665'  20 62                  jr      nz,vardat5
1667'  CD 1B0F'               call    gtoken
166A'  CD 0DF5'               call    logexp        ;expression_1
166D'  3A 0036"               ld      a,(tokcod)
1670'  FE 5D                  cp      ']'
1672'  C2 203D'               jp      nz,experr
1675'  FD 7E FD               ld      a,(iy-3)
1678'  FE 02                  cp      2             ;constant ?
167A'  28 3D                  jr      z,vardat4
167C'  B7                     or      a             ;expression ?
167D'  28 06                  jr      z,vardat1
167F'  21 0DEC'               ld      hl,ldexa
1682'  CD 188F'               call    codgen        ;code generate
1685'                 vardat1:
1685'  E1                     pop     hl
1686'  CD 183D'               call    stoptb        ;store operand table
1689'  21 16E8'               ld      hl,indobj
```

```
168C'   CD 188F'            call    codgen          :code generate
168F'   CD 1856'            call    decopt          ; iy := iy - 3 ; dec(optble)
1692'   CD 1B0F'            call    gtoken
1695'   FE F0               cp      0f0h            ;':=' ?
1697'   20 1A               jr      nz,vardat2
1699'   F1                  pop     af
169A'   FE 03               cp      3               ;data ?
169C'   CA 203D'            jp      z,experr
169F'   CD 1856'            call    decopt          ; iy := iy - 3 ; dec(optble)
16A2'   3E E5               ld      a,0e5h          ;"  PUSH HL  "
16A4'   CD 1953'            call    ptcod1
16A7'   CD 1B0F'            call    gtoken
16AA'   CD 0DCB'            call    expr1           :expression_2
16AD'   11 E177             ld      de,0e177h       ;"  POP  HL  "
16B0'   C3 1964'            jp      ptcod2          ;"  LD   (HL),A "
                    ;
                    ;       var[ exp ]
16B3'               vardat2:
16B3'   F1                  pop     af
16B4'   3E 7E               ld      a,7eh           ;"  LD   A,(HL) "
16B6'   C3 1953'            jp      ptcod1
                    ;
                    ;       var[ const ] := exp
16B9'               vardat4:
16B9'   E1                  pop     hl
16BA'   FD 7E FE            ld      a,(iy-2)
16BD'   85                  add     a,l
16BE'   6F                  ld      l,a
16BF'   30 01               jr      nc,$+3
16C1'   24                    inc   h               ; hl := hl + opr
16C2'   E5                  push    hl
16C3'   CD 1856'            call    decopt          ; iy := iy - 3 ; dec(optble)
16C6'   CD 1B0F'            call    gtoken
                    ;
                    ;       val := exp
16C9'               vardat5:
16C9'   FE F0               cp      0f0h            ;':=' ?
16CB'   20 14               jr      nz,vardat6
16CD'   E1                  pop     hl
16CE'   F1                  pop     af
16CF'   FE 03               cp      3               ;data ?
16D1'   CA 203D'            jp      z,experr
16D4'   E5                  push    hl
16D5'   CD 1B0F'            call    gtoken
16D8'   CD 0DCF'            call    expr2           ;expression
16DB'   E1                  pop     hl
16DC'   3E 32               ld      a,32h           ;"  LD   (nn),A  "
16DE'   C3 196E'            jp      ptcod3
                    ;
                    ;       val
16E1'               vardat6:
16E1'   E1                  pop     hl
16E2'   F1                  pop     af
16E3'   3E 01               ld      a,1
16E5'   C3 183D'            jp      stoptb
                    ;
16E8'   07          indobj: defb    7
16E9'   5F                  ld      e,a
16EA'   16 00               ld      d,0
16EC'   21 B9B9             ld      hl,oprlw
16EF'   19                  add     hl,de
                    ;
                    ;       constant
16F0'               const::
16F0'   CD 19ED'            call    bytcon          ;byte constant
16F3'   18 0A               jr      const6
16F5'               const5:
16F5'   7C                  ld      a,h
16F6'   B7                  or      a               ;check h = 0...+127 ?
```

```
16F7'   C4 1FAB'              call    nz,ilcnor
16FA'   E5                    push    hl
16FB'   CD 1B0F'              call    gtoken
16FE'   E1                    pop     hl
16FF'                 const6:
16FF'   3E 02                 ld      a,2
1701'   C3 183D'              jp      stoptb          ;store operand table
                      ;
                      ;       function call
1704'                 ufuncl::
1704'   21 0000               ld      hl,0
1707'                 ufuncl1:
1707'   22 0078"              ld      (caladr),hl
170A'   01 000C               ld      bc,adrs-identi
170D'   21 0017"              ld      hl,identi
1710'   CD 1AF3'              call    blkpsh          ;block push
1713'   CD 1736'              call    funcl
1716'   01 000C               ld      bc,adrs-identi
1719'   21 0017"              ld      hl,identi
171C'   CD 1B03'              call    blkpop          ;block pop
171F'   2A 0078"              ld      hl,(caladr)
1722'   22 0023"              ld      (adrs),hl
1725'   21 0016"              ld      hl,idetyp
1728'   36 89                 ld      (hl),89h        ;type = undefined function
172A'   C3 0000*              jp      regsym          ;symbol table register
                      ;
172D'                 funcal::
172D'   3A 0016"              ld      a,(idetyp)
1730'   07                    rlca                    ;undefined function ?
1731'   38 D4                 jr      c,ufuncl1
1733'   22 0078"              ld      (caladr),hl
                      ;
                      ;           func( exp_1, ... , exp_n; const_1, const_2 )
1736'                 funcl::
1736'   CD 1B0F'              call    gtoken
1739'   FE 28                 cp      '('
173B'   C2 203D'              jp      nz,experr
173E'   CD 1944'              call    pacod           ;"  PUSH AF  "
1741'   3A 0039"              ld      a,(loopb)
1744'   B7                    or      a               ;loop # ?
1745'   F5                    push    af
1746'   3E C5                 ld      a,0c5h          ;"  PUSH BC  "
1748'   C4 1953'              call    nz,ptcodl
174B'   2A 0078"              ld      hl,(caladr)
174E'   E5                    push    hl
174F'   CD 1B0F'              call    gtoken
1752'   FE 3B                 cp      ';'
1754'   28 4D                 jr      z,funcal4
1756'   FE 29                 cp      ')'
1758'   CA 1808'              jp      z,funcal7
175B'   11 3601               ld      de,3601h
175E'   D5                    push    de
175F'   18 04                 jr      funcal2
1761'                 funcal1:
1761'   D5                    push    de
1762'   CD 1B0F'              call    gtoken
1765'                 funcal2:
1765'   CD 0DCB'              call    exprl           ;expression_n
1768'   CD 1856'              call    decopt          ; iy := iy - 3 ; edc(optble)
176B'   3E F5                 ld      a,0f5h          ;"  PUSH AF  "
176D'   CD 1953'              call    ptcodl
1770'   D1                    pop     de
1771'   3A 0036"              ld      a,(tokcod)
1774'   FE 3B                 cp      ';'
1776'   28 16                 jr      z,funcal3
1778'   FE 29                 cp      ')'
177A'   28 12                 jr      z,funcal3
177C'   FE 2C                 cp      ','
177E'   C2 203D'              jp      nz,experr
```

```
1781'   1C                     inc    e
1782'   7B                     ld     a,e
1783'   FE 21                  cp     32+1
1785'   38 DA                  jr     c,funcal1
1787'   CC 1F5F'               call   z,tmarg
178A'   1E 21                  ld     e,32+1
178C'   18 D3                  jr     funcal1
178E'                  funcal3:
178E'   F5                     push   af
178F'   D5                     push   de
1790'   3E 21                  ld     a,21h
1792'   2A 0008"               ld     hl,(datorg)      ;"  LD   HL,DSEG  "
1795'   CD 196E'               call   ptcod3
1798'   D1                     pop    de
1799'   CD 1964'               call   ptcod2           ;"  LD   (HL),n  "
179C'   F1                     pop    af
179D'   FE 29                  cp     ')'
179F'   28 74                  jr     z,funcal8
17A1'   18 0D                  jr     funcal5
17A3'                  funcal4:
17A3'   3E AF                  ld     a,0afh           ;"  XOR  A  "
17A5'   CD 1953'               call   ptcod1
17A8'   3E 32                  ld     a,32h
17AA'   2A 0008"               ld     hl,(datorg)      ;"  LD   (DSEG),A  "
17AD'   CD 196E'               call   ptcod3
17B0'                  funcal5:
17B0'   CD 1B0F'               call   gtoken
17B3'   FE 2C                  cp     ','
17B5'   28 28                  jr     z,funcal6
17B7'   FE BF                  cp     0bfh             ; ix ?
17B9'   20 10                  jr     nz,funcal5.2
17BB'   11 DDE5                ld     de,0dde5h        ;"  PUSH IX  "
17BE'   CD 1964'               call   ptcod2
17C1'   3E E1                  ld     a,0e1h           ;"  POP  HL  "
17C3'   CD 1953'               call   ptcod1
17C6'   CD 1B0F'               call   gtoken
17C9'   18 08                  jr     funcal5.5
17CB'                  funcal5.2:
17CB'   CD 19C7'               call   conexp           ;constant expression
17CE'   3E 21                  ld     a,21h
17D0'   CD 196E'               call   ptcod3           ;"  LD   HL,nn  "
17D3'                  funcal5.5:
17D3'   3A 0036"               ld     a,(tokcod)
17D6'   FE 29                  cp     ')'
17D8'   28 3B                  jr     z,funcal8
17DA'   FE 2C                  cp     ','
17DC'   C2 203D'               jp     nz,experr
17DF'                  funcal6:
17DF'   CD 1B0F'               call   gtoken
17E2'   FE C0                  cp     0c0h             ; iy ?
17E4'   20 10                  jr     nz,funcal6.2
17E6'   11 FDE5                ld     de,0fde5h        ;"  PUSH IY  "
17E9'   CD 1964'               call   ptcod2
17EC'   3E D1                  ld     a,0d1h           ;"  POP  HL  "
17EE'   CD 1953'               call   ptcod1
17F1'   CD 1B0F'               call   gtoken
17F4'   18 08                  jr     funcal6.5
17F6'                  funcal6.2:
17F6'   CD 19C7'               call   conexp           ;constant expression
17F9'   3E 11                  ld     a,11h
17FB'   CD 196E'               call   ptcod3           ;"  LD   DE,nn  "
17FE'                  funcal6.5:
17FE'   3A 0036"               ld     a,(tokcod)
1801'   FE 29                  cp     ')'
1803'   C2 203D'               jp     nz,experr
1806'   18 0D                  jr     funcal8
1808'                  funcal7:
1808'   3E AF                  ld     a,0afh           ;"  XOR  A  "
```

```
180A'   CD 1953'              call    ptcod1
180D'   3E 32                 ld      a,32h
180F'   2A 0008"              ld      hl,(datorg)     ;"  LD   (DSEG),A  "
1812'   CD 196E'              call    ptcod3
1815'                 funcal8:
1815'   D1                    pop     de
1816'   2A 000C"              ld      hl,(cloc)
1819'   23                    inc     hl
181A'   22 0078"              ld      (caladr),hl     ; funadr := cloc + 1
181D'   EB                    ex      de,hl
181E'   3E CD                 ld      a,0cdh
1820'   CD 196E'              call    ptcod3          ;"  CALL nn  "
1823'   F1                    pop     af
1824'   3E C1                 ld      a,0c1h          ;"  POP  BC  "
1826'   C4 1953'              call    nz,ptcod1
1829'   3E 00                 ld      a,0
182B'   CD 183D'              call    stoptb          ;store operand table
182E'   C3 1B0F'              jp      gtoken
                      ;
                      ;       **  preset operand table pointer & work clear  **
                      ;
1831'                 propt::
1831'   FD 21 003C"           ld      iy,oprtbl
1835'   AF                    xor     a
1836'   32 003B"              ld      (optble),a
1839'   32 003A"              ld      (puhptf),a
183C'   C9                    ret
                      ;
                      ;       **  store operand table  **
                      ;
183D'                 stoptb:: ;      a : operand type ( 0=exp. 1=var. 2=cons )
                               ;      hl: address or constant
183D'   FD 77 00              ld      (iy+0),a
1840'   FD 75 01              ld      (iy+1),l
1843'   FD 74 02              ld      (iy+2),h        ;store
1846'   11 0003               ld      de,3
1849'   FD 19                 add     iy,de
184B'   21 003B"              ld      hl,optble
184E'   34                    inc     (hl)
184F'   7E                    ld      a,(hl)          ;check
1850'   FE 14                 cp      (oprtbe-oprtbl)/3
1852'   D2 203D'              jp      nc,experr
1855'   C9                    ret
                      ;
                      ;       **  decrement operand table pointer  **
                      ;
1856'                 decopt::
1856'   FD 2B                 dec     iy
1858'   FD 2B                 dec     iy
185A'   FD 2B                 dec     iy
185C'   E5                    push    hl
185D'   21 003B"              ld      hl,optble
1860'   35                    dec     (hl)
1861'   E1                    pop     hl
1862'   C9                    ret
                      ;
                      ;       **  exchange operand 1. operand 2  **
                      ;
1863'                 oprexc::
1863'   FD E5                 push    iy
1865'   D1                    pop     de
1866'   21 FFFD               ld      hl,-3
1869'   19                    add     hl,de
186A'   06 03                 ld      b,3
186C'                 oprexc1:
186C'   2B                    dec     hl
186D'   1B                    dec     de
186E'   1A                    ld      a,(de)
```

```
186F'   4E                      ld      c,(hl)
1870'   EB                      ex      de,hl
1871'   71                      ld      (hl),c
1872'   12                      ld      (de),a
1873'   10 F7                   djnz    oprexcl
1875'   C9                      ret
                        ;
                        ;
00B8                    opr1b   equ     0b8h    ;operand 1 byte data mark
B9B9                    opr1w   equ     0b9b9h  ;operand 1 word data mark
00BA                    opr2b   equ     0bah    ;operand 2 byte data mark
BBBB                    opr2w   equ     0bbbbh  ;operand 2 word data mark
                        ;
                        ;       **  dyadic operation, code generate  **
                        ;
1876'                   dyagen:: ;      hl: address table address
                                 ;      iy: operand table pointer
1876'   FD 7E FA                ld      a,(iy-6)
1879'   5F                      ld      e,a
187A'   87                      add     a,a
187B'   83                      add     a,e
187C'   FD 86 FD                add     a,(iy-3)        ; a := m(iy-6)*3 + m(iy-3)
187F'   FE 08                   cp      8               ; a = 8 ?
1881'   CA 192B'                jp      z,codgen10      ;skip if so
1884'   87                      add     a,a
1885'   5F                      ld      e,a
1886'   16 00                   ld      d,0
1888'   19                      add     hl,de
1889'   5E                      ld      e,(hl)
188A'   23                      inc     hl
188B'   56                      ld      d,(hl)
188C'   EB                      ex      de,hl           ; hl = object code store address
188D'   18 0B                   jr      codgen0
                        ;
                        ;       **  code generate  **
                        ;
188F'                   codgen:: ;      hl: object code store address
                                 ;      iy: operand table pointer
188F'   11 0003                 ld      de,3
1892'   FD 19                   add     iy,de           ; iy := iy + 3
1894'   EB                      ex      de,hl
1895'   21 003B"                ld      hl,optble
1898'   34                      inc     (hl)            ; inc(optble)
1899'   EB                      ex      de,hl
189A'                   codgen0:
189A'   7E                      ld      a,(hl)
189B'   47                      ld      b,a
189C'   B7                      or      a               ;"PUSH AF" output ?
189D'   FC 1930'                call    m,codgen15
18A0'   C5                      push    bc
18A1'   E5                      push    hl
18A2'   CD 0000*                call    putobj          ;put object
18A5'   E1                      pop     hl
18A6'   C1                      pop     bc
18A7'                   codgen1:
18A7'   23                      inc     hl
18A8'   C5                      push    bc
18A9'   7E                      ld      a,(hl)
18AA'   FE B8                   cp      opr1b           ;operand 1, byte out ?
18AC'   20 05                   jr      nz,codgen2
18AE'   FD 7E FB                ld      a,(iy-5)
18B1'   18 62                   jr      codgen7
18B3'                   codgen2:
18B3'   FE B9                   cp      low opr1w       ;operand 1, word out ?
18B5'   20 0C                   jr      nz,codgen3
18B7'   FD 7E FB                ld      a,(iy-5)
18BA'   E5                      push    hl
```

```
18BB'   CD 0000*                call    putobj
18BE'   FD 7E FC                ld      a,(iy-4)
18C1'   18 17                   jr      codgen5
18C3'                   codgen3:
18C3'   FE BA                   cp      opr2b           ;operand 2. byte out ?
18C5'   20 05                   jr      nz,codgen4
18C7'   FD 7E FE                ld      a,(iy-2)
18CA'   18 49                   jr      codgen7
18CC'                   codgen4:
18CC'   FE BB                   cp      low opr2w       ;operand 2. word out ?
18CE'   20 1B                   jr      nz,codgen6
18D0'   FD 7E FE                ld      a,(iy-2)
18D3'   E5                      push    hl
18D4'   CD 0000*                call    putobj
18D7'   FD 7E FF                ld      a,(iy-1)
18DA'                   codgen5:
18DA'   CD 0000*                call    putobj
18DD'   2A 000C"                ld      hl,(cloc)
18E0'   23                      inc     hl
18E1'   23                      inc     hl
18E2'   22 000C"                ld      (cloc),hl       ; cloc := cloc + 2
18E5'   E1                      pop     hl
18E6'   23                      inc     hl
18E7'   C1                      pop     bc
18E8'   05                      dec     b
18E9'   18 30                   jr      codgen8
18EB'                   codgen6:
18EB'   FE CD                   cp      0cdh            ;call ?
18ED'   20 26                   jr      nz,codgen7
18EF'   E5                      push    hl
18F0'   CD 0000*                call    putobj
18F3'   E1                      pop     hl
18F4'   23                      inc     hl
18F5'   5E                      ld      e,(hl)
18F6'   23                      inc     hl
18F7'   56                      ld      d,(hl)
18F8'   E5                      push    hl
18F9'   2A 0006"                ld      hl,(codorg)
18FC'   19                      add     hl,de           ;add bias to call address
18FD'   E5                      push    hl
18FE'   7D                      ld      a,l
18FF'   CD 0000*                call    putobj          ;put call hi-address
1902'   F1                      pop     af
1903'   CD 0000*                call    putobj          ;put call low-address
1906'   2A 000C"                ld      hl,(cloc)
1909'   23                      inc     hl
190A'   23                      inc     hl
190B'   23                      inc     hl
190C'   22 000C"                ld      (cloc),hl       ; cloc := cloc + 3
190F'   E1                      pop     hl
1910'   C1                      pop     bc
1911'   05                      dec     b
1912'   05                      dec     b
1913'   18 06                   jr      codgen8
1915'                   codgen7:
1915'   E5                      push    hl
1916'   CD 1959'                call    ptcodx          ;code out
1919'   E1                      pop     hl
191A'   C1                      pop     bc
191B'                   codgen8:
191B'   10 8A                   djnz    codgen1
191D'   FD 36 FA 00             ld      (iy-6),0
1921'   3E FF                   ld      a,0ffh
1923'   32 003A"                ld      (puhptf),a
1926'   CD 1856'                call    decopt          ; iy := iy - 3 ; dec(optble)
1929'   B7                      or      a               ; cy := 0
192A'   C9                      ret
                        ;
```

```
192B'                    codgen10:
192B'   CD 1856'                 call    decopt          ; iy := iy - 3 : dec(optble)
192E'   37                       scf                     ; cy := 1
192F'   C9                       ret
                         ;
1930'                    codgen15:
1930'   ED 44                    neg
1932'   47                       ld      b,a
1933'   3A 003A"                 ld      a,(puhptf)
1936'   B7                       or      a
1937'   78                       ld      a,b
1938'   C8                       ret     z
1939'   3C                       inc     a
193A'   C5                       push    bc
193B'   E5                       push    hl
193C'   CD 1959'                 call    ptcodx          ;put code length
193F'   E1                       pop     hl
1940'   C1                       pop     bc
1941'   3E F5                    ld      a,0f5h          ;"  PUSH AF  "
1943'   C9                       ret
                         ;
                         ;       **  put "  PUSH AF  " code  **
                         ;
1944'                    pacod::
1944'   3A 003A"                 ld      a,(puhptf)
1947'   B7                       or      a
1948'   3E F5                    ld      a,0f5h          ;"  PUSH AF  "
194A'   C2 1953'                 jp      nz,ptcod1
194D'   3E FF                    ld      a,0ffh
194F'   32 003A"                 ld      (puhptf),a
1952'   C9                       ret
                         ;
                         ;       **  put object code  ( 1...3 byte )  **
                         ;
1953'                    ptcod1::                ;  1 byte out  ( reg a )
1953'   F5                       push    af
1954'   AF                       xor     a
1955'   CD 0000*                 call    putobj          ;code length 1 byte
1958'   F1                       pop     af
1959'                    ptcodx:
1959'   CD 0000*                 call    putobj          ;put reg a
195C'   2A 000C"                 ld      hl,(cloc)
195F'   23                       inc     hl
1960'   22 000C"                 ld      (cloc),hl
1963'   C9                       ret
                         ;
1964'                    ptcod2::                ;  2 byte out  ( reg d, e )
1964'   63                       ld      h,e
1965'   6A                       ld      l,d
                         ;
1966'                    ptcodw::                ;  word out  ( reg l, h )
1966'   E5                       push    hl
1967'   3E 02                    ld      a,2
1969'   CD 0000*                 call    putobj          ;code length 2 byte
196C'   18 0B                    jr      ptcody
                         ;
196E'                    ptcod3::                ;  3 byte out  ( reg a, l, h )
196E'   E5                       push    hl
196F'   F5                       push    af
1970'   3E 03                    ld      a,3
1972'   CD 0000*                 call    putobj          ;code length 3 byte
1975'   F1                       pop     af
1976'   CD 1959'                 call    ptcodx          ;put reg a
1979'                    ptcody:
1979'   E1                       pop     hl
197A'   7D                       ld      a,l
197B'   E5                       push    hl
197C'   CD 0000*                 call    putobj          ;put reg l
197F'   F1                       pop     af
```

```
1980'   CD 0000*             call    putobj          ;put reg h
1983'   2A 000C"             ld      hl,(cloc)
1986'   23                   inc     hl
1987'   23                   inc     hl
1988'   22 000C"             ld      (cloc),hl
198B'   C9                   ret
                    ;
                    ;        **  move identifier to name adra  **
                    ;
198C'               movname::

198C'   21 0017"             ld      hl,identi
198F'   11 0026"             ld      de,strbuf
1992'   06 0C                ld      b,adrs-identi
1994'               movnaml:
1994'   7E                   ld      a,(hl)
1995'   12                   ld      (de),a
1996'   B7                   or      a
1997'   C8                   ret     z
1998'   23                   inc     hl
1999'   13                   inc     de
199A'   10 F8                djnz    movnaml
199C'   78                   ld      a,b
199D'   12                   ld      (de),a
199E'   C9                   ret
                    ;
                    ;        **  put name  **
                    ;
199F'               ptname:: ;       a : name type ( 56h...5ah )
199F'   CD 0000*             call    putobj
19A2'   21 0026"             ld      hl,strbuf
19A5'               ptnamel:
19A5'   7E                   ld      a,(hl)
19A6'   B7                   or      a
19A7'   CA 0000*             jp      z,putobj
19AA'   E5                   push    hl
19AB'   CD 0000*             call    putobj
19AE'   E1                   pop     hl
19AF'   23                   inc     hl
19B0'   18 F3                jr      ptnamel
                    ;
                    ;        **  put location count  **
                    ;
19B2'               ptloc::
19B2'   E5                   push    hl
19B3'   3E 80                ld      a,80h
19B5'   18 03                jr      ptchal
                    ;
                    ;        **  put chain address  **
                    ;
19B7'               ptcha::
19B7'   E5                   push    hl
19B8'   3E 81                ld      a,81h
19BA'               ptchal:
19BA'   CD 0000*             call    putobj          ;chain
19BD'   E1                   pop     hl
19BE'   7D                   ld      a,l
19BF'   E5                   push    hl
19C0'   CD 0000*             call    putobj          ;put reg l
19C3'   F1                   pop     af
19C4'   C3 0000*             jp      putobj          ;put reg h
                    ;
                    ;        **  constant expression  **
                    ;
19C7'               conexp::
19C7'   CD 1A01'             call    wodcon          ;word constant
19CA'               conexpl:
19CA'   3A 0036"             ld      a,(tokcod)
19CD'   FE 2B                cp      '+'
```

```
 19CF'   28 11                jr     z,conexp2
 19D1'   FE 2D                cp     '-'
 19D3'   C0                   ret    nz
 19D4'   E5                   push   hl
 19D5'   CD 1B0F'             call   gtoken
 19D8'   CD 1A01'             call   wodcon          ;word constant
 19DB'   EB                   ex     de,hl
 19DC'   E1                   pop    hl
 19DD'   B7                   or     a
 19DE'   ED 52                sbc    hl,de           ; subtract
 19E0'   18 E8                jr     conexp1
 19E2'               conexp2:
 19E2'   E5                   push   hl
 19E3'   CD 1B0F'             call   gtoken
 19E6'   CD 1A01'             call   wodcon          ;word constant
 19E9'   D1                   pop    de
 19EA'   19                   add    hl,de           ; add
 19EB'   18 DD                jr     conexp1
                     ;
                     ;        **  byte constant  **
                     ;
 19ED'               bytcon::
 19ED'   21 0038"             ld     hl,datinl
 19F0'   7E                   ld     a,(hl)
 19F1'   F5                   push   af
 19F2'   36 00                ld     (hl),0
 19F4'   CD 1A01'             call   wodcon
 19F7'   F1                   pop    af
 19F8'   32 0038"             ld     (datinl),a
 19FB'   7C                   ld     a,h
 19FC'   B7                   or     a
 19FD'   C2 1FAB'             jp     nz,ilcnor             ;error : illegal constant
 1A00'   C9                   ret
                     ;
                     ;        **  word constant  **
                     ;
 1A01'               wodcon::
 1A01'   21 0038"             ld     hl,datinl
 1A04'   46                   ld     b,(hl)
 1A05'   3A 0036"             ld     a,(tokcod)
 1A08'   FE BB                cp     0bbh            ;'hi' ?
 1A0A'   28 07                jr     z,bytcon1
 1A0C'   FE BC                cp     0bch            ;'low' ?
 1A0E'   28 0F                jr     z,bytcon2
 1A10'   C3 1A2F'             jp     wodcon0
 1A13'               bytcon1:
 1A13'   C5                   push   bc
 1A14'   36 00                ld     (hl),0
 1A16'   CD 1B0F'             call   gtoken
 1A19'   CD 1A2F'             call   wodcon0
 1A1C'   6C                   ld     l,h
 1A1D'   18 09                jr     bytcon3
 1A1F'               bytcon2:
 1A1F'   C5                   push   bc
 1A20'   36 00                ld     (hl),0
 1A22'   CD 1B0F'             call   gtoken
 1A25'   CD 1A2F'             call   wodcon0
 1A28'               bytcon3:
 1A28'   F1                   pop    af
 1A29'   32 0038"             ld     (datinl),a
 1A2C'   26 00                ld     h,0
 1A2E'   C9                   ret
                     ;
 1A2F'               wodcon0:
 1A2F'   3A 0036"             ld     a,(tokcod)
 1A32'   B7                   or     a
 1A33'   20 05                jr     nz,wodcon1
 1A35'   2A 0012"             ld     hl,(conval)
 1A38'   18 43                jr     wodcon5
```

```
1A3A'                wodcon1:
1A3A'   3D                   dec     a
1A3B'   20 15                jr      nz,wodcon2
1A3D'   21 0016"             ld      hl,idetyp
1A40'   CD 0000*             call    seasym          :search symbol table
1A43'   B7                   or      a               :found ?
1A44'   C4 1E86'             call    nz,miscon
1A47'   3A 0016"             ld      a,(idetyp)
1A4A'   E6 0F                and     0fh
1A4C'   3D                   dec     a               :constant name ?
1A4D'   C2 1FDE'             jp      nz,badcon
1A50'   18 28                jr      wodcon4
1A52'                wodcon2:
1A52'   FE 2D                cp      '.'-1
1A54'   C2 1FDE'             jp      nz,badcon
1A57'   CD 1B0F'             call    gtoken          :get next token
1A5A'   3D                   dec     a
1A5B'   C2 1FF2'             jp      nz,conerr
1A5E'   21 0016"             ld      hl,idetyp
1A61'   CD 0000*             call    seasym          :search symbol table
1A64'   B7                   or      a               :found ?
1A65'   20 49                jr      nz,wodcon8
1A67'   3A 0016"             ld      a,(idetyp)
1A6A'   E6 0F                and     0fh
1A6C'   3D                   dec     a               :constant name ?
1A6D'   CA 1FF2'             jp      z,conerr        :error if so
1A70'   3D                   dec     a               :variable name ?
1A71'   28 07                jr      z,wodcon4
1A73'   3D                   dec     a               :data name ?
1A74'   28 04                jr      z,wodcon4
1A76'   D6 05                sub     5               :lable or funtioc name ?
1A78'   30 09                jr      nc,wodcon6
1A7A'                wodcon4:
1A7A'   2A 0023"             ld      hl,(adrs)
1A7D'                wodcon5:
1A7D'   E5                   push    hl
1A7E'   CD 1B0F'             call    gtoken
1A81'   E1                   pop     hl
1A82'   C9                   ret
1A83'                wodcon6:
1A83'   3A 0038"             ld      a,(datinl)
1A86'   B7                   or      a               :data or inline ?
1A87'   CA 1FF2'             jp      z,conerr        :error if not
1A8A'   ED 5B 0023"          ld      de,(adrs)
1A8E'   3A 0016"             ld      a,(idetyp)
1A91'   4F                   ld      c,a
1A92'   E6 0F                and     0fh
1A94'   FE 09                cp      9               :function name ?
1A96'   38 0C                jr      c,wodcon7
1A98'   CD 1AD6'             call    funchk          :function ?
1A9B'   D2 1FF2'             jp      nc,conerr       :error if lable
1A9E'   CB 79                bit     7,c
1AA0'   20 24                jr      nz,wodcon9
1AA2'   18 D6                jr      wodcon4
1AA4'                wodcon7:
1AA4'   CD 1AD6'             call    funchk          :lable ?
1AA7'   DA 1FF2'             jp      c,conerr        :error if function
1AAA'   CB 79                bit     7,c
1AAC'   20 18                jr      nz,wodcon9
1AAE'   18 CA                jr      wodcon4
1AB0'                wodcon8:
1AB0'   3A 0038"             ld      a,(datinl)
1AB3'   B7                   or      a               :data or inline ?
1AB4'   CA 1FF2'             jp      z,conerr        :error if not
1AB7'   11 0000              ld      de,0
1ABA'   CD 1AD6'             call    funchk          :function ?
1ABD'   3E 98                ld      a,98h           :undefined lable
1ABF'   30 02                jr      nc,$+4
```

```
1AC1'   3E 89              ld      a,89h           ;undefined function
1AC3'   32 0016"           ld      (idetyp),a      ;set type
1AC6'             wodcon9:
1AC6'   2A 000C"           ld      hl,(cloc)
1AC9'   22 0023"           ld      (adrs),hl
1ACC'   D5                 push    de
1ACD'   21 0016"           ld      hl,idetyp
1AD0'   CD 0000*           call    regsym          ;symbol table register
1AD3'   E1                 pop     hl
1AD4'   18 A7              jr      wodcon5
                  ;
1AD6'             funchk:
1AD6'   2A 0000"           ld      hl,(cptr)
1AD9'   CD 1DBE'           call    spskip          ;space skip
1ADC'   28 13              jr      z,funchk1       ;end of line ?
1ADE'   FE 28              cp      '('
1AE0'   20 0F              jr      nz,funchk1
1AE2'   CD 1DC8'           call    nxspsk          ;space skip
1AE5'   28 0A              jr      z,funchk1       ;end of line ?
1AE7'   FE 29              cp      ')'
1AE9'   20 06              jr      nz,funchk1
1AEB'   23                 inc     hl
1AEC'   22 0000"           ld      (cptr),hl
1AEF'   37                 scf                     ; cy := 1
1AF0'   C9                 ret
1AF1'             funchk1:
1AF1'   B7                 or      a               ; cy := 0
1AF2'   C9                 ret
                  ;
                  ;        **  block push  **
                  ;
1AF3'             blkpsh:: ;       bc: push size
                           ;       hl: source address
1AF3'   DD E1              pop     ix
1AF5'   EB                 ex      de,hl
1AF6'   21 0000            ld      hl,0
1AF9'   39                 add     hl,sp
1AFA'   B7                 or      a
1AFB'   ED 42              sbc     hl,bc
1AFD'   F9                 ld      sp,hl
1AFE'   EB                 ex      de,hl
1AFF'   ED B0              ldir                    ;push
1B01'   DD E9              jp      (ix)
                  ;
                  ;        **  block pop  **
                  ;
1B03'             blkpop:: ;       bc:pop size
                           ;       hl:destination address
1B03'   DD E1              pop     ix
1B05'   EB                 ex      de,hl
1B06'   21 0000            ld      hl,0
1B09'   39                 add     hl,sp
1B0A'   ED B0              ldir                    ;pop
1B0C'   F9                 ld      sp,hl
1B0D'   DD E9              jp      (ix)
                  ;
                  ;        **  get next token  **
                  ;
1B0F'             gtoken::
1B0F'   3A 0036"           ld      a,(tokcod)
1B12'   FE FF              cp      0ffh            ;end ?
1B14'   C8                 ret     z               ;return if so
                  ;
1B15'   2A 0000"           ld      hl,(cptr)
1B18'             gtoken0:
1B18'   CD 1DBE'           call    spskip          ;space skip
1B1B'   20 22              jr      nz,gtoken3
1B1D'             gtoken1:
1B1D'   CD 0000*           call    getsou          ;get source program ( 1 line )
```

```
1B20'   30 0A               jr      nc,gtoken2
1B22'               gtoken1.5:
1B22'   CD 0000*            call    edincl          ;end check
1B25'   38 F1               jr      c.gtoken0
1B27'   3E FF               ld      a.0ffh          ; a = 0ffh : end of program
1B29'   C3 1C4B'            jp      gtoken33
1B2C'               gtoken2:
1B2C'   ED 53 0002"         ld      (lino).de
1B30'   E5                  push    hl
1B31'   EB                  ex      de.hl
1B32'   3A 0005"            ld      a,(stmflg)
1B35'   B7                  or      a
1B36'   C4 0000*            call    nz.ddspli       ;line number display object out
1B39'   E1                  pop     hl
1B3A'   CD 1DBE'            call    spskip          ;space skip
1B3D'   28 DE               jr      z,gtoken1
1B3F'               gtoken3:
1B3F'   FE 2F               cp      '/'
1B41'   20 10               jr      nz,gtoken3.5
1B43'   23                  inc     hl
1B44'   7E                  ld      a,(hl)
1B45'   2B                  dec     hl
1B46'   FE 2A               cp      '*'             ;'/*' ?
1B48'   7E                  ld      a,(hl)
1B49'   20 08               jr      nz,gtoken3.5
1B4B'   23                  inc     hl
1B4C'   CD 1D92'            call    coment          ;comment skip
1B4F'   38 D1               jr      c,gtoken1.5
1B51'   18 C5               jr      gtoken0
1B53'               gtoken3.5:
1B53'   22 0014"            ld      (eptr),hl
1B56'   EB                  ex      de,hl
                    ;
                    ;  constant  ( decimal , hexadecimal & character )
                    ;
1B57'   CD 1DCB'            call    numtst          ;numeric ?
1B5A'   38 05               jr      c,gtoken4       ;skip if not
1B5C'   CD 1DF2'            call    decbin         ;convert decimal string into binary value
1B5F'   18 3D               jr      gtoken7
1B61'               gtoken4:
1B61'   FE 24               cp      '$'
1B63'   20 11               jr      nz,gtoken6
1B65'   13                  inc     de
1B66'   1A                  ld      a,(de)
1B67'   CD 1DD2'            call    hextst          ;hexadecimal ?
1B6A'   38 05               jr      c,gtoken5       ;skip if not
1B6C'   CD 1E2A'            call    hexbin          ;convert hex string into binary value
1B6F'   18 2D               jr      gtoken7
1B71'               gtoken5:
1B71'   2A 000C"            ld      hl,(cloc)       ;load code location counter
1B74'   18 28               jr      gtoken7
1B76'               gtoken6:
1B76'   FE 27               cp      ''''
1B78'   20 2B               jr      nz,gtoken10
1B7A'   26 00               ld      h,0
1B7C'   13                  inc     de
1B7D'   1A                  ld      a,(de)
1B7E'   B7                  or      a
1B7F'   28 0A               jr      z.ccher1
1B81'   6F                  ld      l,a             ;load character constant
1B82'   13                  inc     de
1B83'   1A                  ld      a,(de)
1B84'   FE 27               cp      ''''
1B86'   20 0E               jr      nz,ccher2
1B88'   13                  inc     de
1B89'   18 13               jr      gtoken7
1B8B'               ccher1:
1B8B'   21 0027             ld      hl,''''
1B8E'   22 0026"            ld      (strbuf).hl
```

```
1B91'   21 0020          ld      hl,' '
1B94'   18 08            jr      gtoken7
1B96'                cchar2:
1B96'   22 0027"         ld      (strbuf+1),hl
1B99'   3E 27            ld      a,''''
1B9B'   32 0026"         ld      (strbuf),a
1B9E'                gtoken7:
1B9E'   AF               xor     a               ; a = 0 : constant
1B9F'   22 0012"         ld      (conval),hl     ;store value
1BA2'   C3 1C44'         jp      gtoken31.5
                     ;
                     ;  reserved word & identifier
                     ;
1BA5'                gtoken10:
1BA5'   FE 5F            cp      '_'             ;'_' ?
1BA7'   28 05            jr      z,$+7
1BA9'   CD 1DE4'          call   lettst          ;letter ?
1BAC'   38 51             jr     c,gtoken25      ;skip if not
1BAE'   21 0017"         ld      hl,identi
1BB1'   06 0C            ld      b,adrs-identi
1BB3'                gtoken11:
1BB3'   77               ld      (hl),a          ;store identifier area
1BB4'   23               inc     hl
1BB5'                gtoken12:
1BB5'   13               inc     de
1BB6'   1A               ld      a,(de)
1BB7'   CD 1DCB'         call    numtst          ;numeric ?
1BBA'   30 09            jr      nc,gtoken13
1BBC'   CD 1DE4'         call    lettst          ;letter ?
1BBF'   30 04            jr      nc,gtoken13
1BC1'   FE 5F            cp      '_'             ;'_' ?
1BC3'   20 06            jr      nz,gtoken14     ;skip if not
1BC5'                gtoken13:
1BC5'   10 EC            djnz    gtoken11
1BC7'   06 01            ld      b,1
1BC9'   18 EA            jr      gtoken12
1BCB'                gtoken14:
1BCB'   36 00            ld      (hl),0          ;store end of string mark
1BCD'   ED 53 0000"      ld      (cptr),de
                     ;
                     ;  search reserved word
                     ;
1BD1'   11 1C58'         ld      de,rsword
1BD4'                gtoken15:
1BD4'   21 0017"         ld      hl,identi
1BD7'                gtoken16:
1BD7'   1A               ld      a,(de)
1BD8'   B7               or      a
1BD9'   FA 1BEA'         jp      m,gtoken17
1BDC'   28 1D            jr      z,gtoken20
1BDE'   4F               ld      c,a
1BDF'   7E               ld      a,(hl)
1BE0'   CD 1DB5'         call    traupc          ;translate to upper case
1BE3'   B9               cp      c
1BE4'   20 0C            jr      nz,gtoken18
1BE6'   13               inc     de
1BE7'   23               inc     hl
1BE8'   18 ED            jr      gtoken16
1BEA'                gtoken17:
1BEA'   7E               ld      a,(hl)
1BEB'   B7               or      a
1BEC'   20 0A            jr      nz,gtoken19
1BEE'   1A               ld      a,(de)          ; a = reserved word No.
1BEF'   C3 1C4B'         jp      gtoken33
1BF2'                gtoken18:
1BF2'   13               inc     de
1BF3'   1A               ld      a,(de)
1BF4'   B7               or      a
1BF5'   F2 1BF2'         jp      p,gtoken18
```

```
1BF8'                   gtoken19:
1BF8'   13                      inc     de
1BF9'   18 D9                   jr      gtoken15
1BFB'                   gtoken20:
1BFB'   3E 01                   ld      a,1             ; a = 1 : identifier
1BFD'   18 4C                   jr      gtoken33
                        ;
                        ;  search special character ( delimiter.operator, ... )
                        ;
1BFF'                   gtoken25:
1BFF'   21 1D77'                ld      hl,specha
1C02'   01 001B                 ld      bc,speche-specha
1C05'   ED B1                   cpir                    ;search
1C07'   20 46                   jr      nz,gtoken35
1C09'   4F                      ld      c,a
1C0A'   13                      inc     de
1C0B'   ED 53 0000"             ld      (cptr),de
                        ;
1C0F'   FE 3A                   cp      ':'
1C11'   20 07                   jr      nz,gtoken27
1C13'   1A                      ld      a,(de)
1C14'   FE 3D                   cp      '='             ;':=' ?
1C16'   3E F0                   ld      a,0f0h
1C18'   18 27                   jr      gtoken30
1C1A'                   gtoken27:
1C1A'   FE 3C                   cp      '<'
1C1C'   20 0B                   jr      nz,gtoken28
1C1E'   1A                      ld      a,(de)
1C1F'   D6 3C                   sub     '<'             ;'<<' or '<=' or '<>' ?
1C21'   FE 03                   cp      3
1C23'   30 25                   jr      nc,gtoken32
1C25'   C6 F1                   add     a,0f1h
1C27'   18 1A                   jr      gtoken31
1C29'                   gtoken28:
1C29'   FE 3E                   cp      '>'
1C2B'   20 0B                   jr      nz,gtoken29
1C2D'   1A                      ld      a,(de)
1C2E'   D6 3D                   sub     '='             ;'>=' or '>>' ?
1C30'   FE 02                   cp      2
1C32'   30 16                   jr      nc,gtoken32
1C34'   C6 F4                   add     a,0f4h
1C36'   18 0B                   jr      gtoken31
1C38'                   gtoken29:
1C38'   FE 40                   cp      '@'
1C3A'   20 0E                   jr      nz,gtoken32
1C3C'   1A                      ld      a,(de)
1C3D'   FE 40                   cp      '@'             ;'@@' ?
1C3F'   3E F6                   ld      a,0f6h
1C41'                   gtoken30:
1C41'   20 07                   jr      nz,gtoken32
1C43'                   gtoken31:
1C43'   13                      inc     de
1C44'                   gtoken31.5:
1C44'   ED 53 0000"             ld      (cptr),de
1C48'   18 01                   jr      gtoken33
1C4A'                   gtoken32:
1C4A'   79                      ld      a,c
1C4B'                   gtoken33:
1C4B'   32 0036"                ld      (tokcod),a
1C4E'   C9                      ret
                        ;
1C4F'                   gtoken35:
1C4F'   6F                      ld      l,a
1C50'   26 00                   ld      h,0
1C52'   22 0026"                ld      (strbuf),hl
1C55'   C3 1F79'                jp      illchr
                        ;
                        ;       << reserved word >>
```

```
                              ;
1C58'                         rsword::
1C58'   41 4E 44 A8                   defb    'AND'.          0a8h
1C5C'   41 54 BE                      defb    'AT',           0beh
1C5F'   42 52 45 41                   defb    'BREAK',        08bh
1C63'   4B 8B
1C65'   42 59 9A                      defb    'BY',           09ah
1C68'   42 59 54 45                   defb    'BYTE',         091h
1C6C'   91
1C6D'   43 41 52 52                   defb    'CARRY',        0b4h
1C71'   59 B4
1C73'   43 4F 4E 53                   defb    'CONS',         08ch
1C77'   8C
1C78'   44 41 54 41                   defb    'DATA',         08eh
1C7C'   8E
1C7D'   44 45 43 AD                   defb    'DEC',          0adh
1C81'   44 45 43 4A                   defb    'DECJ',         0b3h
1C85'   B3
1C86'   44 45 58 A5                   defb    'DEX',          0a5h
1C8A'   44 45 42 55                   defb    'DEBUG',        084h
1C8E'   47 84
1C90'   45 4C 53 45                   defb    'ELSE',         095h
1C94'   95
1C95'   45 4C 53 45                   defb    'ELSEIF'.       0bdh
1C99'   49 46 BD
1C9C'   45 58 49 54                   defb    'EXIT',         09eh
1CA0'   9E
1CA1'   46 4F 52 98                   defb    'FOR'.          098h
1CA5'   47 4F 9C                      defb    'GO',           09ch
1CA8'   47 4F 54 4F                   defb    'GOTO'.         09dh
1CAC'   9D
1CAD'   48 49 BB                      defb    'HI',           0bbh
1CB0'   49 46 93                      defb    'IF',           093h
1CB3'   49 4E 43 AC                   defb    'INC',          0ach
1CB7'   49 4E 43 4C                   defb    'INCLUDE',      081h
1CBB'   55 44 45 81
1CBF'   49 4E 4C 49                   defb    'INLINE',       090h
1CC3'   4E 45 90
1CC6'   49 4E 58 A4                   defb    'INX'.          0a4h
1CCA'   49 58 BF                      defb    'IX',           0bfh
1CCD'   49 59 C0                      defb    'IY'.           0c0h
1CD0'   4C 44 58 A2                   defb    'LDX'.          0a2h
1CD4'   4C 4F 4F 50                   defb    'LOOP',         09bh
1CD8'   9B
1CD9'   4C 4F 57 BC                   defb    'LOW'.          0bch
1CDD'   4D 45 4D 4F                   defb    'MEMORY'.       0b9h
1CE1'   52 59 B9
1CE4'   4D 49 4E 55                   defb    'MINUS'.        0abh
1CE8'   53 AB
1CEA'   4E 4F 54 A9                   defb    'NOT',          0a9h
1CEE'   4F 52 A7                      defb    'OR',           0a7h
1CF1'   4F 56 45 52                   defb    'OVERFLOW'.     0b8h
1CF5'   46 4C 4F 57
1CF9'   B8
1CFA'   50 41 52 49                   defb    'PARITY'.       0b7h
1CFE'   54 59 B7
1D01'   50 4C 55 53                   defb    'PLUS'.         0aah
1D05'   AA
1D06'   50 4F 52 54                   defb    'PORT'.         0bah
1D0A'   BA
1D0B'   50 52 4F 47                   defb    'PROG',         080h
1D0F'   80
1D10'   52 45 43 55                   defb    'RECURSIVE',    08fh
1D14'   52 53 49 56
1D18'   45 8F
1D1A'   52 45 54 55                   defb    'RETURN'.       09fh
1D1E'   52 4E 9F
1D21'   52 4C AE                      defb    'RL',           0aeh
```

```
1D24'   52 4C 43 AF              defb    'RLC',          0afh
1D28'   52 52 B0                 defb    'RR',           0b0h
1D2B'   52 52 43 B1              defb    'RRC',          0b1h
1D2F'   53 45 54 A1              defb    'SET',          0a1h
1D33'   53 49 47 4E              defb    'SIGN',         0b6h
1D37'   B6
1D38'   53 52 41 B2              defb    'SRA',          0b2h
1D3C'   53 54 4F 50              defb    'STOP',         0a0h
1D40'   A0
1D41'   53 54 58 A3              defb    'STX',          0a3h
1D45'   54 48 45 4E              defb    'THEN',         094h
1D49'   94
1D4A'   54 4F 99                 defb    'TO',           099h
1D4D'   54 52 4F 46              defb    'TROFF',        086h
1D51'   46 86
1D53'   54 52 4F 4E              defb    'TRON',         085h
1D57'   85
1D58'   55 4E 54 49              defb    'UNTIL',        097h
1D5C'   4C 97
1D5E'   56 41 52 8D              defb    'VAR',          08dh
1D62'   57 48 49 4C              defb    'WHILE',        096h
1D66'   45 96
1D68'   57 4F 52 44              defb    'WORD',         092h
1D6C'   92
1D6D'   58 4F 52 A6              defb    'XOR',          0a6h
1D71'   5A 45 52 4F              defb    'ZERO',         0b5h
1D75'   B5
1D76'   00                       defb    0
                         ;
                         ;       <<  special character  >>
                         ;
1D77'                    specha::
1D77'   21 22 23 25              defb    '!"#%&()*+,-./:;'
1D7B'   26 28 29 2A
1D7F'   2B 2C 2D 2E
1D83'   2F 3A 3B
1D86'   3C 3D 3E 3F              defb    '<=>?@[]^{|}~'
1D8A'   40 5B 5D 5E
1D8E'   7B 7C 7D 7E
1D92'                    speche  equ     $
                         ;
                         ;       **  comment skip  **
                         ;
1D92'                    coment::
1D92'   23                       inc     hl
1D93'                    coment1:
1D93'   7E                       ld      a,(hl)
1D94'   B7                       or      a
1D95'   28 14                    jr      z,coment2
1D97'   FE 2A                    cp      '*'
1D99'   20 F7                    jr      nz,coment
1D9B'   23                       inc     hl
1D9C'   7E                       ld      a,(hl)
1D9D'   B7                       or      a
1D9E'   28 0B                    jr      z,coment2
1DA0'   FE 2A                    cp      '*'
1DA2'   28 EF                    jr      z,coment1
1DA4'   FE 2F                    cp      '/'
1DA6'   20 EA                    jr      nz,coment
1DA8'   B7                       or      a               ;cy := 0
1DA9'   23                       inc     hl
1DAA'   C9                       ret
1DAB'                    coment2:
1DAB'   CD 0000*                 call    getsou          ;get source program
1DAE'   D8                       ret     c               ;return if cy = 1
1DAF'   ED 53 0002"              ld      (lino),de
1DB3'   18 DE                    jr      coment1
                         ;
                         ;       **  translate to upper case  **
```

```
                                ;
1DB5'                           traupc:: ;      a : character
1DB5'   FE 61                           cp      'a'
1DB7'   D8                              ret     c
1DB8'   FE 7B                           cp      'z'+1
1DBA'   D0                              ret     nc
1DBB'   D6 20                           sub     20h
1DBD'   C9                              ret
                                ;
                                ;       **  space skip  **
                                ;
1DBE'                           spskip:: ;      hl : string address
1DBE'   7E                              ld      a,(hl)
1DBF'   B7                              or      a               ;end of line ?
1DC0'   C8                              ret     z               ;return if so ( eol : z=1 )
1DC1'   FE 09                           cp      09h             ;tab code ?
1DC3'   28 03                           jr      z,nxspsk        ;skip if so
1DC5'   FE 20                           cp      ' '             ;space ?
1DC7'   C0                              ret     nz              ;return if not ( non space : z=0 )
1DC8'                           nxspsk::
1DC8'   23                              inc     hl
1DC9'   18 F3                           jr      spskip
                                ;
                                ;       **  numeric test  **
                                ;
1DCB'                           numtst:: ;      a : character
1DCB'   FE 30                           cp      '0'
1DCD'   D8                              ret     c
1DCE'   FE 3A                           cp      '9'+1
1DD0'   3F                              ccf                     ; cy = 1 : non numeric
1DD1'   C9                              ret                     ; cy = 0 : numerec
1DD2'                           hextst:: ;      a : character
1DD2'   CD 1DCB'                        call    numtst
1DD5'   D0                              ret     nc
1DD6'   FE 41                           cp      'A'
1DD8'   D8                              ret     c
1DD9'   FE 47                           cp      'F'+1
1DDB'   3F                              ccf
1DDC'   D0                              ret     nc
1DDD'   FE 61                           cp      'a'
1DDF'   D8                              ret     c
1DE0'   FE 67                           cp      'f'+1
1DE2'   3F                              ccf                     ; cy = 1 : non hexadecimal
1DE3'   C9                              ret                     ; cy = 0 : hexadecimal
                                ;
                                ;       **  letter test  **
                                ;
1DE4'                           lettst:: ;      a : character
1DE4'   FE 41                           cp      'A'
1DE6'   D8                              ret     c
1DE7'   FE 5B                           cp      'Z'+1
1DE9'   3F                              ccf
1DEA'   D0                              ret     nc
1DEB'   FE 61                           cp      'a'
1DED'   D8                              ret     c
1DEE'   FE 7B                           cp      'z'+1
1DF0'   3F                              ccf                     ; cy = 1 : non letter
1DF1'   C9                              ret                     ; cy = 0 : letter
                                ;
                                ;       **  convert decimal string into binary value  **
                                ;
1DF2'                           decbin:: ;      de : string address
1DF2'   21 0000                         ld      hl,0
1DF5'                           decbin1:
1DF5'   1A                              ld      a,(de)
1DF6'   D6 30                           sub     '0'             ; a := m(de) - '0'
1DF8'   FE 0A                           cp      9+1             ;numeric ?
1DFA'   D0                              ret     nc              ;return if not ( hl : value )
1DFB'   44                              ld      b,h
```

```
1DFC'   4D                      ld      c,l
1DFD'   29                      add     hl,hl
1DFE'   38 12                   jr      c,dcoverr
1E00'   29                      add     hl,hl
1E01'   38 0F                   jr      c,dcoverr
1E03'   09                      add     hl,bc
1E04'   38 0C                   jr      c,dcoverr
1E06'   29                      add     hl,hl           ; hl := hl * 10
1E07'   38 09                   jr      c,dcoverr
1E09'   4F                      ld      c,a
1E0A'   06 00                   ld      b,0
1E0C'   09                      add     hl,bc           ; hl := hl + a
1E0D'   38 03                   jr      c,dcoverr
1E0F'   13                      inc     de
1E10'   18 E3                   jr      decbinl
                        ;
1E12'                   dcoverr:
1E12'   2A 0014"                ld      hl,(eptr)
1E15'   11 0026"                ld      de,strbuf
1E18'   06 0F                   ld      b,strbfe-strbuf-1
1E1A'                   dcverl:
1E1A'   7E                      ld      a,(hl)
1E1B'   CD 1DCB'                call    numtst
1E1E'   38 49                   jr      c,hcver3
1E20'   05                      dec     b
1E21'   04                      inc     b
1E22'   28 03                   jr      z,dcver2
1E24'   12                      ld      (de),a
1E25'   13                      inc     de
1E26'   05                      dec     b
1E27'                   dcver2:
1E27'   23                      inc     hl
1E28'   18 F0                   jr      dcverl
                        ;
                        ;       **  convert hex string into binary value  **
                        ;
1E2A'                   hexbin:: ;      de : string
1E2A'   21 0000                 ld      hl,0
1E2D'                   hexbinl:
1E2D'   1A                      ld      a,(de)
1E2E'   CD 1DD2'                call    hextst          ;hexadecimal ?
1E31'   D8                      ret     c               ;return if not ( hl : value )
1E32'   CD 1DB5'                call    traupc          ;translate to upper case
1E35'   D6 30                   sub     '0'
1E37'   FE 0A                   cp      9+1
1E39'   38 02                   jr      c,hexbin2
1E3B'   D6 07                   sub     'A'-'9'-1
1E3D'                   hexbin2:
1E3D'   4F                      ld      c,a
1E3E'   06 00                   ld      b,0
1E40'   7C                      ld      a,h
1E41'   E6 F0                   and     0f0h
1E43'   20 08                   jr      nz,hcoverr
1E45'   29                      add     hl,hl
1E46'   29                      add     hl,hl
1E47'   29                      add     hl,hl
1E48'   29                      add     hl,hl           ; hl := hl << 4
1E49'   09                      add     hl,bc
1E4A'   13                      inc     de
1E4B'   18 E0                   jr      hexbinl
                        ;
1E4D'                   hcoverr:
1E4D'   2A 0014"                ld      hl,(eptr)
1E50'   11 0026"                ld      de,strbuf
1E53'   06 0E                   ld      b,strbfe-strbuf-2
1E55'   7E                      ld      a,(hl)
1E56'   23                      inc     hl
1E57'   12                      ld      (de),a          ;move '$' character
1E58'   13                      inc     de
```

```
1E59'                   hcver1:
1E59'   7E                      ld      a,(hl)
1E5A'   CD 1DD2'                call    hextst
1E5D'   38 0A                   jr      c,hcver3
1E5F'   05                      dec     b
1E60'   04                      inc     b
1E61'   28 03                   jr      z,hcver2
1E63'   12                      ld      (de),a
1E64'   13                      inc     de
1E65'   05                      dec     b
1E66'                   hcver2:
1E66'   23                      inc     hl
1E67'   18 F0                   jr      hcver1
1E69'                   hcver3:
1E69'   E5                      push    hl
1E6A'   AF                      xor     a
1E6B'   12                      ld      (de),a
1E6C'   CD 1FBB'                call    illcon
1E6F'   D1                      pop     de
1E70'   21 0000                 ld      hl,0
1E73'   C9                      ret
                        ;
                        ;       **  error  **
                        ;

                        ;
                        ; missing variable name
                        ;
1E74'                   misvar::
1E74'   11 1E7D'                ld      de,mvar
1E77'   CD 1E98'                call    ermisn
1E7A'   36 02                   ld      (hl),2
1E7C'   C9                      ret
1E7D'   76 61 72 69     mvar:   defb    'variable',0
1E81'   61 62 6C 65
1E85'   00
                        ;
                        ; missing constant name
                        ;
1E86'                   miscon::
1E86'   11 1E8F'                ld      de,mcon
1E89'   CD 1E98'                call    ermisn
1E8C'   36 01                   ld      (hl),1
1E8E'   C9                      ret
1E8F'   63 6F 6E 73     mcon:   defb    'constant',0
1E93'   74 61 6E 74
1E97'   00
                        ;
1E98'                   ermisn:
1E98'   21 0000                 ld      hl,0
1E9B'   22 0023"                ld      (adrs),hl
1E9E'   21 1EA8'                ld      hl,mserm
1EA1'   CD 0000*                call    error
1EA4'   21 0016"                ld      hl,idetyp
1EA7'   C9                      ret
1EA8'   23 4D 69 73     mserm:  defb    '#Missing ¥ name : @'
1EAC'   73 69 6E 67
1EB0'   20 5C 20 6E
1EB4'   61 6D 65 20
1EB8'   3A 20 40
1EBB'   0017"                   defw    identi
1EBD'   00                      defb    0
                        ;
                        ; bad option switch
                        ;
1EBE'                   opserr::
1EBE'   21 1EC4'                ld      hl,opserm
1EC1'   C3 2053'                jp      abt
```

```
1EC4'   23 42 61 64   opserm: defb    '#Bad option switch',0
1EC8'   20 6F 70 74
1ECC'   69 6F 6E 20
1ED0'   73 77 69 74
1ED4'   63 68 00
                      ;
                      ; illegal function name
                      ;
1ED7'                 ilferr::
1ED7'   21 1EDD'              ld      hl,ilferm
1EDA'   C3 2053'              jp      abt
1EDD'   23            ilferm: defb    '#'
1EDE'   49 6C 6C 65   illfun: defb    'Illegal function name',0
1EE2'   67 61 6C 20
1EE6'   66 75 6E 63
1EEA'   74 69 6F 6E
1EEE'   20 6E 61 6D
1EF2'   65 00
                      ;
1EF4'                 ilefun::
1EF4'   21 1EFD'              ld      hl,ilefum
1EF7'   11 0017"              ld      de,identi
1EFA'   C3 0000*              jp      error
1EFD'   23 40         ilefum: defb    '#@'
1EFF'   1EDE'                 defw    illfun
1F01'   20 3A 20 5C           defb    ' : ¥',0
1F05'   00
                      ;
                      ; illegal name
                      ;
1F06'                 ilnerr::
1F06'   21 1F0C'              ld      hl,ilnerm
1F09'   C3 2053'              jp      abt
1F0C'   23            ilnerm: defb    '#'
1F0D'   49 6C 6C 65   illnam: defb    'Illegal name',0
1F11'   67 61 6C 20
1F15'   6E 61 6D 65
1F19'   00
                      ;
1F1A'                 ilenam::
1F1A'   21 1F23'              ld      hl,ilenmm
1F1D'   11 0017"              ld      de,identi
1F20'   C3 0000*              jp      error
1F23'   23 40         ilenmm: defb    '#@'
1F25'   1F0D'                 defw    illnam
1F27'   20 3A 20 5C           defb    ' : ¥',0
1F2B'   00
                      ;
                      ; illegal label
                      ;
1F2C'                 ilelab::
1F2C'   11 0017"              ld      de,identi
1F2F'   21 1F35'              ld      hl,illbem
1F32'   C3 0000*              jp      error
1F35'   23 49 6C 6C   illbem: defb    '#Illegal label : ¥',0
1F39'   65 67 61 6C
1F3D'   20 6C 61 62
1F41'   65 6C 20 3A
1F45'   20 5C 00
                      ;
                      ; bad string data
                      ;
1F48'                 bstrdt::
1F48'   21 1F4E'              ld      hl,bstrms
1F4B'   C3 0000*              jp      error
1F4E'   23 42 61 64   bstrms: defb    '#Bad string data',0
1F52'   20 73 74 72
1F56'   69 6E 67 20
1F5A'   64 61 74 61
```

```
1F5E'   00
                        ;
                        ; too many arguments
                        ;
1F5F'                   tmarg::
1F5F'   21 1F65'                ld      hl,tmargm
1F62'   C3 0000*                jp      error
1F65'   23 54 6F 6F     tmargm: defb    '#Too many arguments',0
1F69'   20 6D 61 6E
1F6D'   79 20 61 72
1F71'   67 75 6D 65
1F75'   6E 74 73 00
                        ;
                        ; illegal character
                        ;
1F79'                   illchr::
1F79'   21 1F82'                ld      hl,illchm
1F7C'   11 0026"                ld      de,strbuf
1F7F'   C3 2053'                jp      abt
1F82'   23 49 6C 6C     illchm: defb    '#Illegal character : ¥',0
1F86'   65 67 61 6C
1F8A'   20 63 68 61
1F8E'   72 61 63 74
1F92'   65 72 20 3A
1F96'   20 5C 00
                        ;
                        ; illegal constant
                        ;
1F99'                   ilcnds::
1F99'   11 1F9E'                ld      de,ilcndm
1F9C'   18 20                   jr      illcon1
1F9E'   64 69 73 70     ilcndm: defb    'displacement',0
1FA2'   6C 61 63 65
1FA6'   6D 65 6E 74
1FAA'   00
                        ;
1FAB'                   ilcnor::
1FAB'   11 1FB0'                ld      de,ilcnom
1FAE'   18 0E                   jr      illcon1
1FB0'   6F 76 65 72     ilcnom: defb    'over range',0
1FB4'   20 72 61 6E
1FB8'   67 65 00
                        ;
1FBB'                   illcon:
1FBB'   11 0026"                ld      de,strbuf
1FBE'                   illcon1:
1FBE'   21 1FC8'                ld      hl,illcnm
1FC1'   CD 0000*                call    error
1FC4'   21 0000                 ld      hl,0
1FC7'   C9                      ret
1FC8'   23 49 6C 6C     illcnm: defb    '#Illegal constant : ¥',0
1FCC'   65 67 61 6C
1FD0'   20 63 6F 6E
1FD4'   73 74 61 6E
1FD8'   74 20 3A 20
1FDC'   5C 00
                        ;
                        ; bad constant
                        ;
1FDE'                   badcon::
1FDE'   21 1FE4'                ld      hl,badcnm
1FE1'   C3 2053'                jp      abt
1FE4'   23 42 61 64     badcnm: defb    '#Bad constant',0
1FE8'   20 63 6F 6E
1FEC'   73 74 61 6E
1FF0'   74 00
                        ;
                        ; bad address constant
                        ;
```

```
1FF2'                   conerr::
1FF2'   21 1FF8'                ld      hl,bdadcm
1FF5'   C3 2053'                jp      abt
1FF8'   23 42 61 64     bdadcm: defb    '#Bad address constant',0
1FFC'   20 61 64 64
2000'   72 65 73 73
2004'   20 63 6F 6E
2008'   73 74 61 6E
200C'   74 00
                        ;
                        ; syntax error
                        ;
200E'                   synerr::
200E'   21 2014'                ld      hl,syems
2011'   C3 2053'                jp      abt
2014'   23 53 79 6E     syems:  defb    '#Syntax error',0
2018'   74 61 78 20
201C'   65 72 72 6F
2020'   72 00
                        ;
                        ; bad index operation
                        ;
2022'                   idxerr::
2022'   21 2028'                ld      hl,idems
2025'   C3 2053'                jp      abt
2028'   23 42 61 64     idems:  defb    '#Bad index operation',0
202C'   20 69 6E 64
2030'   65 78 20 6F
2034'   70 65 72 61
2038'   74 69 6F 6E
203C'   00
                        ;
                        ; bad expression
                        ;
203D'                   experr::
203D'   21 2043'                ld      hl,exems
2040'   C3 2053'                jp      abt
2043'   23 42 61 64     exems:  defb    '#Bad expression',0
2047'   20 65 78 70
204B'   72 65 73 73
204F'   69 6F 6E 00
                        ;
                        ;       abort
2053'                   abt:
2053'   CD 0000*                call    error
2056'   C3 0000*                jp      abort


                        ;-----------------------------------------------
                        ;-              w o r k   a r e a              -
                        ;-----------------------------------------------
2059'                           dseg

0000"                   cptr::          defs    2       ;character pointer ( source program )
0002"                   lino::          defs    2       ;source line number
0004"                   optisw::        defs    1       ;option switch
0005"                   stmflg::        defs    1       ;not 00h: statement, 00h: non statement

0006"                   codorg::        defs    2       ;code segment origin address
0008"                   datorg::        defs    2       ;data segment origin address
000A"                   stkbot::        defs    2       ;stack bottom address + 1

000C"                   cloc::          defs    2       ;code location counter
000E"                   dloc::          defs    2       ;data location counter
0010"                   wloc::          defs    2       ;work data location counter
0012"                   conval::        defs    2       ;constant value

0014"                   eptr::          defs    2       ;error string pointer
```

```
0016"       idetyp::    defs   1       ;identifier type
0017"       identi::    defs   12      ;identifier ( max 12 character )
0023"       adrs::      defs   2       ;address ( & constant value )
0025"                   defs   1       ;filler

0026"       strbuf::    defs   16      ;string buffer
0036"       strbfe      equ    $

0036"       tokcod::    defs   1      ;token code
0037"       localf::    defs   1      ; 00h: global, not 00h: local
0038"       datinl::    defs   1      ; 01h: data declare, 02h: inline statement
0039"       loopb::     defs   1      ;not 00h: use loop # statement
003A"       puhptf::    defs   1      ;not 00h: object " PUSH AF " output

003B"       optble::    defs   1       ;operand table store length
003C"       oprtbl::    defs   3*20    ;operand table
0078"       oprtbe      equ    $

0078"       caladr::    defs   2       ;function & subroutine call address

007A"       lopadr::    defs   2       ;loop address
007C"       extadr::    defs   2       ;exit address
007E"       convar::    defs   2       ;control variable address
0080"       terpar::    defs   2       ;terminal parameter address
0082"       incpar::    defs   2       ;incrementation parameter address
0084"       skpadr::    defs   2       ;skip address
0086"       nxtadr::    defs   2       ;next address

0088"       varadr::    defs   2       ;variable address
008A"       varsiz::    defs   2       ;variable size

008C"       idxpsf::    defs   1       ;index push flag
008D"       retjad::    defs   2       ;return statement jump address

008F"       work.e::

                   end
```

(3)　STRUNTIM. MACの構成とアセンブル・リスト

STRUNTIM. MACはStellarのランタイム・ルーチンで，先頭に13個のエントリ・ポイントがあります。各エントリ処理は次のようになっています。ここでTOPとはランタイム・ルーチンの先頭アドレスのことです。TOPのアドレスは表意定数__CODEで得られます。

@start (TOP+0)：生成したオブジェクト・プログラムを実行する。

@stop (TOP+3)：オブジェクト・プログラムの実行を終わらせる。

@mul.m (TOP+6)：１バイトの乗算（レジスタＡ←レジスタＡ＊レジスタHLが示すメモリの値)。

@mul.r (TOP+9)：１バイトの乗算（レジスタＡ←レジスタＡ＊レジスタＤ)。

@div.m (TOP+12)：１バイトの除算（レジスタＡ←レジスタＡ／レジスタHLが示すメモリの値)。

@div.r (TOP+15)：１バイトの除算(レジスタＡ←レジスタＡ／レジスタＤ)。

@rem.m (TOP+18)：１バイトの剰余(レジスタＡ←レジスタＡ％レジスタHLが示すメモリの値)。

@rem.r (TOP+21)：１バイトの剰余（レジスタＡ←レジスタＡ％レジスタＤ)。

@shl.m (TOP+24)：左シフト（レジスタＡ←レジスタＡ＜＜レジスタＨＬが示すメモリの値)。

@shl.r (TOP+27)：左シフト(レジスタＡ←レジスタＡ＜＜レジスタＤ)。

@shr.m (TOP+30)：右シフト（レジスタＡ←レジスタＡ＞＞レジスタＨＬが示すメモリの値)。

@shr.r (TOP+33)：右シフト（レジスタＡ←レジスタＡ＞＞レジスタＤ)。

@setarg (TOP+36)：関数への実引数を仮引数へ転送する。また，再帰的な関数の場合は局所的な変数のセーブも行う。

また，ランタイム・ルーチンはワーク・エリアとして48バイトのRAM領域を使用します。そのうち，次の二つの領域はStellarのプログラムで参照可能な領域です。ここで__workは表意定数__workのことです。

arglen（__work+0）：実引数の数。

actarg（__work+1～32）：実引数の値（第1引数～第32引数まで）。

●リスト2-6　STERUNTIMのアセンブル・リスト

```
                                title   Stellar run time routine  Rev 1.0   05/15/1984
                                name    ('runtim')
                        .z80
                        ;**********************************************
                        ;*                                            *
                        ;*      Stellar run time routine  ( Rev 1.0 ) *
                        ;*                                            *
                        ;*                          05/15/1984        *
                        ;*                                            *
                        ;*        Copyright (c) 1984 H.Ohnuki / MIA   *
                        ;*                                            *
                        ;**********************************************

                        ;---------------------------------------------
                        ;-                 run time                  -
                        ;---------------------------------------------

0000'                                   cseg

                        ;
                        ;       **  run time entry  **
                        ;
0000'   C3 00F7'        @start::        jp      main    ;exec main program
0003'   C3 00DF'        @stop::         jp      stop    ;stop run
0006'   C3 0027'        @mul.m::        jp      mul.m   ;multiply  a := a * m(hl)
0009'   C3 0029'        @mul.r::        jp      mul.r   ;multiply  a := a * d
000C'   C3 0039'        @div.m::        jp      div.m   ;divide    a := a / m(hl)
000F'   C3 003E'        @div.r::        jp      div.r   ;divide    a := a / d
0012'   C3 0043'        @rem.m::        jp      rem.m   ;remainder   a := a % m(hl)
0015'   C3 0044'        @rem.r::        jp      rem.r   ;remainder   a := a % d
0018'   C3 0054'        @shl.m::        jp      shl.m   ;shift left  a := a << m(hl)
001B'   C3 0055'        @shl.r::        jp      shl.r   ;shift left  a := a << d
001E'   C3 0065'        @shr.m::        jp      shr.m   ;shift right  a := a >> m(hl)
0021'   C3 0066'        @shr.r::        jp      shr.r   ;shift right  a := a >> d
0024'   C3 0077'        @setag::        jp      setag   ;set argument

                        ;
                        ;       **  multiply  **
                        ;
                        ; entry  @mul.m  ....  a := a * m(hl)
                        ; entry  @mul.r  ....  a := a * d
                        ;
0027'                   mul.m:
0027'   5E                      ld      e,(hl)
0028'   FE                      defb    0feh
0029'                   mul.r:
0029'   5A                      ld      e,d
```

```
002A'   16 00               ld      d,0
002C'   6A                  ld      l,d
002D'   67                  ld      h,a
002E'   3E 08               ld      a,8
0030'               mul.1:
0030'   29                  add     hl,hl
0031'   30 01               jr      nc,mul.2
0033'   19                  add     hl,de
0034'               mul.2:
0034'   3D                  dec     a
0035'   20 F9               jr      nz,mul.1
0037'   7D                  ld      a.l
0038'   C9                  ret
                    ;
                    ;       **  divide  **
                    ;
                    ; entry  @div.m  ....  a := a / m(hl)
                    ; entry  @div.r  ....  a := a / d
                    ;
0039'               div.m:
0039'   CD 0043'    @1:     call    rem.m
003C'   7B                  ld      a,e
003D'   C9                  ret
003E'               div.r:
003E'   CD 0044'    @2:     call    rem.r
0041'   7B                  ld      a,e
0042'   C9                  ret
                    ;
                    ;       **  remainder  **
                    ;
                    ; entry  @rem.m  ....  a := a % m(hl)
                    ; entry  @rem.r  ....  a := a % d
                    ;
0043'               rem.m:
0043'   56                  ld      d,(hl)
0044'               rem.r:
0044'   5F                  ld      e.a
0045'   AF                  xor     a
0046'   2E 08               ld      l.8
0048'               rem.1:
0048'   CB 23               sla     e
004A'   17                  rla
004B'   BA                  cp      d
004C'   38 02               jr      c,rem.2
004E'   92                  sub     d
004F'   1C                  inc     e
0050'               rem.2:
0050'   2D                  dec     l
0051'   20 F5               jr      nz.rem.l
0053'   C9                  ret
                    ;
                    ;       **  shift left  **
                    ;
                    ; entry  @shl.m  ....  a := a << m(hl)
                    ; entry  @shl.r  ....  a := a << d
                    ;
0054'               shl.m:
0054'   56                  ld      d.(hl)
0055'               shl.r:
0055'   14                  inc     d
0056'   15                  dec     d
0057'   C8                  ret     z
0058'   5F                  ld      e.a
0059'   7A                  ld      a.d
005A'   FE 09               cp      9
005C'   3E 00               ld      a,0
005E'   D0                  ret     nc
005F'   7B                  ld      a,e
0060'               shl.1:
```

```
0060'   87                      add     a,a
0061'   15                      dec     d
0062'   20 FC                   jr      nz,shl.1
0064'   C9                      ret
                        ;
                        ;       **  shift right  **
                        ;
                        ; entry  @shr.m  ....  a := a >> m(hl)
                        ; entry  @shr.r  ....  a := a >> d
                        ;
0065'                   shr.m:
0065'   56                      ld      d,(hl)
0066'                   shr.r:
0066'   14                      inc     d
0067'   15                      dec     d
0068'   C8                      ret     z
0069'   5F                      ld      e,a
006A'   7A                      ld      a,d
006B'   FE 09                   cp      9
006D'   3E 00                   ld      a,0
006F'   D0                      ret     nc
0070'   7B                      ld      a,e
0071'                   shr.1:
0071'   CB 3F                   srl     a
0073'   15                      dec     d
0074'   20 FB                   jr      nz,shr.1
0076'   C9                      ret


                        ;
                        ;    **  push local variable & move actual argument to dummy argument  **
                        ;
                        ; entry      @setag
                        ; parameter c : dummy  argument length ( 0...32 )
                        ;           de: local variable area size ( reg de = 0 : non recursive )
                        ;           hl: local variable area address ( dummy argument address )
                        ;
0077'                   setag:
0077'   22 0026"        ?1:     ld      (pshadr),hl
007A'   ED 53 0024"     ??2:    ld      (pshsiz),de
007E'   E1                      pop     hl
007F'   22 0028"        ?3:     ld      (retad1),hl
0082'   E1                      pop     hl
0083'   22 002A"        ?4:     ld      (retad2),hl
0086'   3A 0000"        ?5:     ld      a,(arglen)
0089'   B7                      or      a
008A'   28 0D                   jr      z,setag2
008C'   21 0000"        ?6:     ld      hl,actarg-1
008F'   5F                      ld      e,a
0090'   16 00                   ld      d,0
0092'   19                      add     hl,de
0093'   47                      ld      b,a
0094'                   setag1:
0094'   F1                      pop     af
0095'   77                      ld      (hl),a
0096'   2B                      dec     hl
0097'   10 FB                   djnz    setag1
0099'                   setag2:
0099'   2A 002A"        ?7:     ld      hl,(retad2)
009C'   E5                      push    hl
009D'   2A 0024"        ?8:     ld      hl,(pshsiz)
00A0'   7C                      ld      a,h
00A1'   B5                      or      l
00A2'   28 1E                   jr      z,setag3
00A4'   EB                      ex      de,hl
00A5'   21 0000                 ld      hl,0
00A8'   ED 52                   sbc     hl,de
00AA'   39                      add     hl,sp
00AB'   F9                      ld      sp,hl
```

```
00AC'   C5                          push    bc
00AD'   42                          ld      b,d
00AE'   4B                          ld      c,e
00AF'   EB                          ex      de,hl
00B0'   2A 0026"        ?9:         ld      hl,(pshadr)
00B3'   ED B0                       ldir
00B5'   C1                          pop     bc
00B6'   2A 0026"        ?10:        ld      hl,(pshadr)
00B9'   E5                          push    hl
00BA'   2A 0024"        ?11:        ld      hl,(pshsiz)
00BD'   E5                          push    hl
00BE'   21 00D5'        @3:         ld      hl,poplv
00C1'   E5                          push    hl
00C2'                   setag3:
00C2'   79                          ld      a,c
00C3'   B7                          or      a
00C4'   28 0B                       jr      z,setag4
00C6'   2A 0026"        ?12:        ld      hl,(pshadr)
00C9'   EB                          ex      de,hl
00CA'   21 0001"        ?13:        ld      hl,actarg
00CD'   06 00                       ld      b,0
00CF'   ED B0                       ldir
00D1'                   setag4:
00D1'   2A 0028"        ?14:        ld      hl,(retadl)
00D4'   E9                          jp      (hl)
                        ;
                        ;           **  pop local variable  **
                        ;
00D5'                   poplv:
00D5'   C1                          pop     bc
00D6'   D1                          pop     de
00D7'   21 0000                     ld      hl,0
00DA'   39                          add     hl,sp
00DB'   ED B0                       ldir
00DD'   F9                          ld      sp,hl
00DE'   C9                          ret

                        ;
                        ;           **  stop run  **
                        ;
00DF'                   stop:
00DF'   3A 0021"        ?15:        ld      a,(@stptp)
00E2'   B7                          or      a               ;halt ?
00E3'   20 03                       jr      nz,stop2
00E5'                   stopl:
00E5'   76                          halt
00E6'   18 FD                       jr      stopl
00E8'                   stop2:
00E8'   3D                          dec     a               ;return ?
00E9'   20 05                       jr      nz,stop3
00EB'   ED 7B 0022"     ??16:       ld      sp,(@retad)
00EF'   C9                          ret
00F0'                   stop3:
00F0'   3D                          dec     a               ;jump ?
00F1'   20 F2                       jr      nz,stopl
00F3'   2A 0022"        ?17:        ld      hl,(@retad)
00F6'   E9                          jp      (hl)

                        ;
                        ;           **  beginning of main routine  **
                        ;
00F7'                   main:

00F7'                   @prog::     ;---------------------------------------;
                                    ;                                       ;
                                    ;       stellar object code area        ;
                                    ;                                       ;
                                    ;---------------------------------------;
```

```
                          ;
                          ;       **  run time work area  **
                          ;
00F7'                                   dseg
0000"                     @rtwork::
0000"                     arglen: defs    1         ;actual argument length
0001"                     actarg: defs    32        ;actual argument data
0021"                     @stptp::defs    1         ;stop type ( 0:HALT, 1:RET, 2:JP nnnn )
0022"                     @retad::defs    2         ;stop type 1 : entry stack pointer value
                                                    ;stop type 2 : return address
0024"                     pshsiz: defs    2         ;push size
0026"                     pshadr: defs    2         ;push address
0028"                     retad1: defs    2         ;return address
002A"                     retad2: defs    2
002C"                     @temp:: defs    4         ;temporary

                          ;
                          ;       **  beginning of variable  **
                          ;

0030"                     @var::  ;--------------------------------------;
                                  ;                                      ;
                                  ;      global & local variable area    ;
                                  ;                                      ;
                                  ;--------------------------------------;

                          ;----------------------------------------------
                          ;-               relocate address table      -
                          ;----------------------------------------------

0030"                                   cseg
                          ;
                          ; code segment
                          ;
00F7'                     crelat::
00F7'   0001' 0004'               defw    @start+1,@stop+1
00FB'   0007' 000A'               defw    @mul.m+1,@mul.r+1
00FF'   000D' 0010'               defw    @div.m+1,@div.r+1
0103'   0013' 0016'               defw    @rem.m+1,@rem.r+1
0107'   0019' 001C'               defw    @shl.m+1,@shl.r+1
010B'   001F' 0022'               defw    @shr.m+1,@shr.r+1
010F'   0025' 003A'               defw    @setag+1,@1+1
0113'   003F' 00BF'               defw    @2+1,@3+1
0117'   0000                      defw    0
                          ;
                          ; data segment
                          ;
0119'                     drelat::
0119'   0078' 007C'               defw    ?1+1,??2+2
011D'   0080' 0084'               defw    ?3+1,?4+1
0121'   0087' 008D'               defw    ?5+1,?6+1
0125'   009A' 009E'               defw    ?7+1,?8+1
0129'   00B1' 00B7'               defw    ?9+1,?10+1
012D'   00BB' 00C7'               defw    ?11+1,?12+1
0131'   00CB' 00D2'               defw    ?13+1,?14+1
0135'   00E0' 00ED'               defw    ?15+1,??16+2
0139'   00F4'                     defw    ?17+1
013B'   0000                      defw    0

                          ;----------------------------------------------

                                  end
```

〔3〕CONVOBJ. COM

CONVOBJ. COM は CONVOBJ. MAC を次の手順でアセンブル，リンクしてつくります。

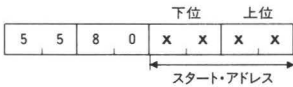
```
M80 = CONVOBJ
L 80 CONVOBJ, CONVOBJ/N/E
```

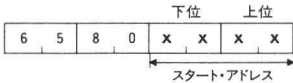
CONVOBJ は Stellar コンパイラが出力したオブジェクト・ファイル（図2-7）を入力し，インテル HEX 形式のファイル（図2-8）を出力します。

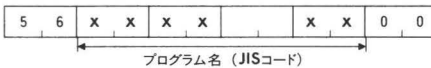
この CONVOBJ. MAC は CP／M用なので，他のパソコン(OS も含む)にコンパイラを移植するときは移植するパソコン(OS) に合わせて入出力などをつくり直す必要があります。

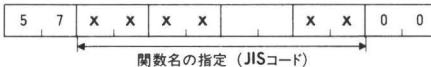
図2-7 Stellar コンパイラが出力するオブジェクト・ファイルの形式

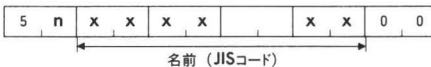
●オブジェクト・スタート（オブジェクト・ファイルは必ずこのオブジェクト・スタートで始まる）

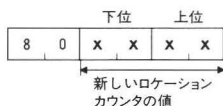
5 5	8 0	x x（下位）	x x（上位）
		スタート・アドレス	

ノーマル・モードでコンパイルしたオブジェクト・ファイル，ロケーション・カウンタへスタート・アドレスを設定する。

6 5	8 0	x x（下位）	x x（上位）
		スタート・アドレス	

デバッグ・モードでコンパイルしたオブジェクト・ファイル，ロケーション・カウンタへスタート・アドレスを設定する。

●プログラム名の指定

現在のロケーション・カウンタの値よりプログラムが始まることを示す。

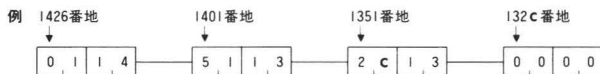
●関数名の指定

現在のロケーション・カウンタの値より関数が始まることを示す。

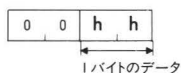
●全域的な名前の指定

現在のロケーション・カウンタの値が名前の値(アドレス)になる。
n＝8～Aで，n＝8なら定数名，n＝9なら変数名，n＝Aならデータ名。

●ロケーション・カウンタの設定

ロケーション・カウンタの値を変更する。

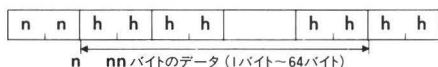
●アドレスのチェイン

チェイン・アドレスで結ばれるすべてのアドレスを現在のロケーション・カウンタの値に置き換える。チェインの最後は2バイトのゼロとする。

例

これを現ロケーション・カウンタの値を142cとしてアドレスのチェイン(81, 26, 14)を行うと次のようになる。

●ストア・データ

ロケーション・カウンタが示すメモリに1バイトの値**hh**をストアする。 ロケーション・カウンタはストア後+1される。

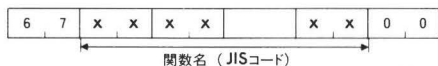
n n h h h h　h h h h

n　nnバイトのデータ(1バイト～64バイト)

ロケーション・カウンタが示すメモリより(1バイト～64バイト) **hh**……**hh**の値を順にストアする。ロケーション・カウンタはストアした分だけプラスされる。

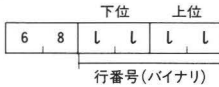
●行番号の表示(CP/Mバージョンではサポートしていない)

トレースのための行番号を表示する。この形式のオブジェクトを入力した場合，次のような命令に展開する。

66 ──→ **CALL** 行番号表示のためのルーチン
ll
ll ──→ **DEFB** ll, ll

●関数名の表示(CP/Mバージョンではサポートしていない)

トレースのための関数名を表示する。この形式のオブジェクトを入力した場合，次のような命令に展開する。

67 ──→ **CALL** 関数名表示のためのルーチン
xx
xx ──→ **DEFB** xx, xx, …, xx, 0
⋮
xx
00

●**実行の中断**（CP/M バージョンではサポートしていない）

実行の中断を行う。この形式のオブジェクトを入力した場合は，次のような命令に展開する。

68 —— CALL 実行中断のためのルーチン
ll
ll —— DEFB ll, ll

●**オブジェクト・エンド**

F F

オブジェクト・ファイルの終わりを示す。

図2-8 インテルHEX形式のファイルの内容

テキスト形のファイルで一つのレコードはCR，LFコードで区切られている。
レコードの形式は次のようになっている。

例

:10010000C3F701C3DF01C32701C32901C33901C3F9

レコード長 16バイト　ロケーション・アドレス 0100番地　レコードタイプ00　データ16バイト分　チェック・サム

:00000001FF ← 最終レコード。最終レコードはレコード長，ロケーション・アドレスともにゼロにする。

●リスト2-7　CONVOBJのアセンブル・リスト

```
                                title   convert object to intel HEX
                                subttl  Rev 1.00  06/15/1984
                                name    ('covobj')
                        .z80
                        ;*********************************************
                        ;*                                           *
                        ;*      convert stellar compiler object.     *
                        ;*              to  intel HEX file           *
                        ;*                                           *
                        ;*                ( Rev 1.00 )  06/15/1984   *
                        ;*                                           *
                        ;*        Copyright (c) 1984 H.Ohnuki / MIA  *
                        ;*                                           *
                        ;*********************************************

 0100                   rev     equ     100h    ;revision 1.00

                        ;
                        ;       cp/m-80,msx-dos interface
                        ;
 0080                   reclen  equ     128     ; 1 record = 128 byte

 0000                   boot    equ     0000h   ;system reboot entry
 0005                   bdos    equ     0005h   ;bdos entry
 005C                   dfcb    equ     005ch   ;default fcb
 0080                   dbuff   equ     0080h   ;default buffer
 0100                   tpa     equ     0100h   ;tpa address

 0009                   htab    equ     09h     ;horizontal tabulation
 000D                   cr      equ     0dh     ;carriage return
 000A                   lf      equ     0ah     ;line feed
 001A                   eof     equ     1ah     ;end of file

 0002                   putchrf equ     2       ;put console character
 0009                   printf  equ     9       ;print string function
 000F                   openf   equ     15      ;open file function
 0010                   closef  equ     16      ;close file function
 0013                   deletf  equ     19      ;delete file function
 0014                   readf   equ     20      ;read next record
 0015                   writef  equ     21      ;write next record
 0016                   makef   equ     22      ;make file
 001A                   setdmf  equ     26      ;set dma address


                        ;---------------------------------------------
                        ;-                      program              -
                        ;---------------------------------------------

 0000'                          cseg

 0000'                  start::
 0000'   2A 0006                ld      hl,(bdos+1)
 0003'   2E 00                  ld      l,0
 0005'   F9                     ld      sp,hl           ;set sp reg
 0006'   11 FF80                ld      de,-128         ;stack size = 128 byte
 0009'   19                     add     hl,de
 000A'   22 0009"               ld      (freeda),hl     ;set free area ending  address
 000D'   23                     inc     hl
 000E'   23                     inc     hl
 000F'   22 000B"               ld      (objmov),hl
 0012'   2A 05C7'               ld      hl,($memry)
 0015'   2E 00                  ld      l,0
 0017'   24                     inc     h
 0018'   22 0007"               ld      (frebga),hl     ;set free area beginning address
 001B'   11 003D'               ld      de,stamsg
 001E'   0E 09                  ld      c,printf
 0020'   CD 0005                call    bdos            ;print starting message
```

```
                                 ;
0023'   3A 005D                          ld      a,(dfcb+1)
0026'   FE 20                            cp      ' '             ;file name ok ?
0028'   CA 03C5'                         jp      z,nofile
002B'   21 0065                          ld      hl,dfcb+9
002E'   7E                               ld      a,(hl)
002F'   FE 20                            cp      ' '
0031'   20 6B                            jr      nz,rdobjf
0033'   36 4F                            ld      (hl),'O'
0035'   23                               inc     hl
0036'   36 42                            ld      (hl),'B'
0038'   23                               inc     hl
0039'   36 4A                            ld      (hl),'J'        ;set file type '.OBJ'
003B'   18 61                            jr      rdobjf
                                 ;
003D'   0D 0A 53 74              stamsg: defb    cr,lf,'Stellar utility,  convert object ==> intel HEX'
0041'   65 6C 6C 61
0045'   72 20 75 74
0049'   69 6C 69 74
004D'   79 2C 20 20
0051'   63 6F 6E 76
0055'   65 72 74 20
0059'   6F 62 6A 65
005D'   63 74 20 3D
0061'   3D 3E 20 69
0065'   6E 74 65 6C
0069'   20 48 45 58
006D'   0D 0A 52 65                      defb    cr,lf,'Rev '
0071'   76 20
0073'   31 2E 30 30                      defb    rev/100h+'0','.',(rev/10h and 0fh)+'0',(rev and 0fh)+'0'
0077'   20 20 43 6F                      defb    '  Copyright (c) 1984  H.Ohnuki / MIA'
007B'   70 79 72 69
007F'   67 68 74 20
0083'   28 63 29 20
0087'   31 39 38 34
008B'   20 20 48 2E
008F'   4F 68 6E 75
0093'   6B 69 20 2F
0097'   20 4D 49 41
009B'   0D 0A 24                 crlf:   defb    cr,lf,'$'
                                 ;
                                 ;       read object file
                                 ;
009E'                            rdobjf::
009E'   AF                               xor     a
009F'   32 000D"                         ld      (symotf),a
00A2'   32 007C                          ld      (dfcb+32),a     ;reset dfcb.cr
00A5'   11 005C                          ld      de,dfcb
00A8'   0E 0F                            ld      c,openf
00AA'   CD 0005                          call    bdos            ;open object file
00AD'   3C                               inc     a
00AE'   CA 03C5'                         jp      z,nofile
00B1'   3E 80                            ld      a,reclen
00B3'   32 0000"                         ld      (bufptr),a
                                 ;
00B6'   CD 0443'                         call    getobj          ;get object file
00B9'   FE 55                            cp      55h             ;start mark ?
00BB'   28 08                            jr      z,rdobjf1
00BD'   FE 65                            cp      65h             ;sym file out ?
00BF'   C2 0422'                         jp      nz,badobj
00C2'   CD 050F'                         call    mksyfl          ;make sym file
00C5'                            rdobjf1:
00C5'   CD 0443'                         call    getobj
00C8'   D6 80                            sub     80h             ;location counter set ?
00CA'   C2 0422'                         jp      nz,badobj
00CD'   C3 020E'                         jp      setlocl
                                 ;
```

```
00D0'                   rdnext:
00D0'   CD 0443'                call    getobj
00D3'   FE FF                   cp      0ffh            ;end mark ?
00D5'   CA 0293'                jp      z,endobj
00D8'   FE 56                   cp      56h
00DA'   DA 020A'                jp      c,setloc
00DD'   FE 5B                   cp      5ah+1           ;name ?
00DF'   D2 020A'                jp      nc,setloc
00E2'   D6 56                   sub     56h
00E4'   87                      add     a,a
00E5'   6F                      ld      l,a
00E6'   26 00                   ld      h,0
00E8'   11 0167'                ld      de,namadt
00EB'   19                      add     hl,de
00EC'   5E                      ld      e,(hl)
00ED'   23                      inc     hl
00EE'   56                      ld      d,(hl)
00EF'   0E 09                   ld      c,printf
00F1'   CD 0005                 call    bdos
                        ;
00F4'   3A 000D"                ld      a,(symotf)
00F7'   B7                      or      a
00F8'   28 25                   jr      z,prtnam1
00FA'   3A 0006"                ld      a,(loc+1)       ;symbol file address out
00FD'   CD 04F9'                call    binhex          ;convert to hex
0100'   D5                      push    de
0101'   7A                      ld      a,d
0102'   CD 054B'                call    wrsymf
0105'   D1                      pop     de
0106'   7B                      ld      a,e
0107'   CD 054B'                call    wrsymf
010A'   3A 0005"                ld      a,(loc)
010D'   CD 04F9'                call    binhex          ;convert to hex
0110'   D5                      push    de
0111'   7A                      ld      a,d
0112'   CD 054B'                call    wrsymf
0115'   D1                      pop     de
0116'   7B                      ld      a,e
0117'   CD 054B'                call    wrsymf
011A'   3E 20                   ld      a,' '
011C'   CD 054B'                call    wrsymf
                        ;
011F'                   prtnam1:
011F'   CD 0443'                call    getobj
0122'   B7                      or      a
0123'   28 14                   jr      z,prtnam3
0125'   5F                      ld      e,a
0126'   3A 000D"                ld      a,(symotf)
0129'   B7                      or      a
012A'   28 06                   jr      z,prtnam2
012C'   D5                      push    de
012D'   7B                      ld      a,e
012E'   CD 054B'                call    wrsymf          ;sym file out
0131'   D1                      pop     de
0132'                   prtnam2:
0132'   0E 02                   ld      c,putchrf
0134'   CD 0005                 call    bdos            ;print name
0137'   18 E6                   jr      prtnam1
0139'                   prtnam3:
0139'   3A 000D"                ld      a,(symotf)
013C'   B7                      or      a
013D'   28 17                   jr      z,prtnam5
013F'   21 002F"                ld      hl,symhcu
0142'   34                      inc     (hl)
0143'   7E                      ld      a,(hl)
0144'   FE 04                   cp      4
0146'   3E 09                   ld      a,htab
0148'   20 09                   jr      nz,prtnam4
014A'   36 00                   ld      (hl),0
```

```
014C'   3E 0D                   ld      a,cr
014E'   CD 054B'                call    wrsymf
0151'   3E 0A                   ld      a,lf
0153'                   prtnam4:
0153'   CD 054B'                call    wrsymf
0156'                   prtnam5:
0156'   11 0206'                ld      de,eqsym
0159'   0E 09                   ld      c,printf
015B'   CD 0005                 call    bdos
015E'   2A 0005"                ld      hl,(loc)
0161'   CD 04E4'                call    prhex4          ;print address
0164'   C3 00D0'                jp      rdnext
                        ;
0167'   0171' 0184'     namadt: defw    prgnam,funnam,connam,varnam,datnam
016B'   0197' 01AA'
016F'   01BD'
0171'   0D 0A 50 72     prgnam: defb    cr,lf,'Program  name : $'
0175'   6F 67 72 61
0179'   6D 20 20 6E
017D'   61 6D 65 20
0181'   3A 20 24
0184'   0D 0A 46 75     funnam: defb    cr,lf,'Function name : $'
0188'   6E 63 74 69
018C'   6F 6E 20 6E
0190'   61 6D 65 20
0194'   3A 20 24
0197'   0D 0A 43 6F     connam: defb    cr,lf,'Constant name : $'
019B'   6E 73 74 61
019F'   6E 74 20 6E
01A3'   61 6D 65 20
01A7'   3A 20 24
01AA'   0D 0A 56 61     varnam: defb    cr,lf,'Variable name : $'
01AE'   72 69 61 62
01B2'   6C 65 20 6E
01B6'   61 6D 65 20
01BA'   3A 20 24
01BD'   0D 0A 44 61     datnam: defb    cr,lf,'Data     name : $'
01C1'   74 61 20 20
01C5'   20 20 20 6E
01C9'   61 6D 65 20
01CD'   3A 20 24
01D0'   0D 0A 0A 45     endadr: defb    cr,lf,lf,'End address   : $'
01D4'   6E 64 20 61
01D8'   64 64 72 65
01DC'   73 73 20 20
01E0'   20 3A 20 24
01E4'   0D 0A 50 72     prgsiz: defb    cr,lf,'Program size  : $'
01E8'   6F 67 72 61
01EC'   6D 20 73 69
01F0'   7A 65 20 20
01F4'   3A 20 24
01F7'   20 20 5B 20     pagsz1: defb    '  [ $'
01FB'   24
01FC'   20 50 61 67     pagsz2: defb    ' Page ]',cr,lf,'$'
0200'   65 20 5D 0D
0204'   0A 24
0206'   20 3D 20 24     eqsym:  defb    ' = $'
                        ;
020A'                   setloc::
020A'   FE 80                   cp      80h             ;location counter set ?
020C'   20 22                   jr      nz,adrcha
020E'                   setloc1:
020E'   F5                      push    af
020F'   CD 0443'                call    getobj
0212'   E5                      push    hl
0213'   CD 0443'                call    getobj
0216'   E1                      pop     hl
0217'   67                      ld      h,a
```

```
0218'   22 0005"                ld      (loc),hl        ;set loc
021B'   F1                      pop     af
021C'   B7                      or      a
021D'   C2 00D0'                jp      nz,rdnext
0220'   22 0003"                ld      (prgorg),hl
0223'   EB                      ex      de,hl
0224'   2A 0007"                ld      hl,(frebga)
0227'   B7                      or      a
0228'   ED 52                   sbc     hl,de
022A'   22 0001"                ld      (offset),hl
022D'   C3 00D0'                jp      rdnext
                        ;
0230'                   adrcha::
0230'   FE 81                   cp      81h             ;address chain ?
0232'   20 21                   jr      nz,codobj
0234'   CD 0443'                call    getobj
0237'   E5                      push    hl
0238'   CD 0443'                call    getobj
023B'   D1                      pop     de
023C'   57                      ld      d,a
023D'   ED 4B 0005"             ld      bc,(loc)
0241'                   adrchal:
0241'   2A 0001"                ld      hl,(offset)
0244'   19                      add     hl,de
0245'   CD 027A'                call    adrchk          ;address check
0248'   5E                      ld      e,(hl)
0249'   23                      inc     hl
024A'   56                      ld      d,(hl)
024B'   70                      ld      (hl),b          ;address set
024C'   2B                      dec     hl
024D'   71                      ld      (hl),c
024E'   7A                      ld      a,d
024F'   B3                      or      e
0250'   20 EF                   jr      nz,adrchal
0252'   C3 00D0'                jp      rdnext
                        ;
0255'                   codobj:
0255'   FE 41                   cp      40h+1
0257'   D2 0422'                jp      nc,badobj
025A'   47                      ld      b,a
025B'   B7                      or      a
025C'   20 01                   jr      nz,codobjl
025E'   04                      inc     b
025F'                   codobjl:
025F'   C5                      push    bc
0260'   CD 0443'                call    getobj
0263'   C1                      pop     bc
0264'   2A 0005"                ld      hl,(loc)
0267'   23                      inc     hl
0268'   22 0005"                ld      (loc),hl
026B'   2B                      dec     hl
026C'   ED 5B 0001"             ld      de,(offset)
0270'   19                      add     hl,de
0271'   CD 027A'                call    adrchk          ;address check
0274'   77                      ld      (hl),a          ;store object code
0275'   10 E8                   djnz    codobjl
0277'   C3 00D0'                jp      rdnext
                        ;
027A'                   adrchk::
027A'   E5                      push    hl
027B'   ED 5B 0007"             ld      de,(frebga)
027F'   B7                      or      a
0280'   ED 52                   sbc     hl,de           ; hl >= frebga ?
0282'   DA 0422'                jp      c,badobj
0285'   E1                      pop     hl
0286'   E5                      push    hl
0287'   ED 5B 0009"             ld      de,(freeda)
028B'   B7                      or      a
028C'   ED 52                   sbc     hl,de           ; hl < freeda ?
```

```
028E'  D2 0422'            jp      nc,badobj
0291'  E1                  pop     hl
0292'  C9                  ret
                   ;
0293'              endobj::
0293'  3A 000D"            ld      a,(symotf)
0296'  B7                  or      a
0297'  C4 0593'            call    nz,clsymf       ;sym file close
029A'  11 01D0'            ld      de,endadr
029D'  0E 09               ld      c,printf
029F'  CD 0005             call    bdos
02A2'  2A 0005"            ld      hl,(loc)
02A5'  2B                  dec     hl
02A6'  CD 04E4'            call    prhex4          ;print end address
02A9'  11 01E4'            ld      de,prgsiz
02AC'  0E 09               ld      c,printf
02AE'  CD 0005             call    bdos
02B1'  2A 0005"            ld      hl,(loc)
02B4'  ED 5B 0003"         ld      de,(prgorg)
02B8'  B7                  or      a
02B9'  ED 52               sbc     hl,de
02BB'  E5                  push    hl
02BC'  CD 04E4'            call    prhex4          ;print program size
02BF'  11 01F7'            ld      de,pagsz1
02C2'  0E 09               ld      c,printf
02C4'  CD 0005             call    bdos
02C7'  E1                  pop     hl
02C8'  11 00FF             ld      de,0ffh
02CB'  19                  add     hl,de
02CC'  6C                  ld      l,h
02CD'  26 00               ld      h,0
02CF'  CD 04A4'            call    prdec           ;print page size
02D2'  11 01FC'            ld      de,pagsz2
02D5'  0E 09               ld      c,printf
02D7'  CD 0005             call    bdos
                   ;
                   ;       write hex file
                   ;
02DA'              wrhexf::
02DA'  21 0065             ld      hl,dfcb+9
02DD'  36 48               ld      (hl),'H'
02DF'  23                  inc     hl
02E0'  36 45               ld      (hl),'E'
02E2'  23                  inc     hl
02E3'  36 58               ld      (hl),'X'        ;set file type '.HEX'
02E5'  23                  inc     hl
02E6'  36 00               ld      (hl),0          ;reset dfcb.ex
02E8'  11 005C             ld      de,dfcb
02EB'  D5                  push    de
02EC'  0E 13               ld      c,deletf
02EE'  CD 0005             call    bdos            ;delete old hex file
02F1'  D1                  pop     de
02F2'  0E 16               ld      c,makef
02F4'  CD 0005             call    bdos            ;make new hex file
02F7'  3C                  inc     a
02F8'  CA 03DD'            jp      z,dirful
02FB'  AF                  xor     a
02FC'  32 007C             ld      (dfcb+32),a     ;reset dfcb.cr
02FF'  32 0000"            ld      (bufptr),a
                   ;
0302'  ED 5B 0003"         ld      de,(prgorg)
0306'              wrhexf1:
0306'  2A 0005"            ld      hl,(loc)
0309'  B7                  or      a
030A'  ED 52               sbc     hl,de           ; de = loc ?
030C'  28 54               jr      z,wrhexf3
030E'  06 10               ld      b,16
0310'  7C                  ld      a,h
```

```
0311'  B7                       or      a
0312'  20 05                    jr      nz,wrhexf1.5
0314'  7D                       ld      a,l
0315'  B8                       cp      b               ; de-loc >= 16 ?
0316'  30 01                    jr      nc,wrhexf1.5
0318'  45                       ld      b,l
0319'                   wrhexf1.5:
0319'  D5                       push    de
031A'  C5                       push    bc
031B'  3E 3A                    ld      a,':'
031D'  CD 047B'                 call    wrhfl           ;write record mark ':'
0320'  F1                       pop     af
0321'  F5                       push    af
0322'  CD 0471'                 call    wrhex           ;write record length (16 byte)
0325'  F1                       pop     af
0326'  D1                       pop     de
0327'  F5                       push    af
0328'  D5                       push    de
0329'  7A                       ld      a,d
032A'  CD 0471'                 call    wrhex
032D'  D1                       pop     de
032E'  D5                       push    de
032F'  7B                       ld      a,e
0330'  CD 0471'                 call    wrhex           ;write location address
0333'  AF                       xor     a
0334'  CD 0471'                 call    wrhex           ;write record type (00h)
0337'  D1                       pop     de
0338'  F1                       pop     af
0339'  47                       ld      b,a
033A'  82                       add     a,d
033B'  83                       add     a,e
033C'  4F                       ld      c,a
033D'                   wrhexf2:
033D'  2A 0001"                 ld      hl,(offset)
0340'  19                       add     hl,de
0341'  7E                       ld      a,(hl)
0342'  81                       add     a,c
0343'  4F                       ld      c,a
0344'  7E                       ld      a,(hl)
0345'  13                       inc     de
0346'  C5                       push    bc
0347'  D5                       push    de
0348'  CD 0471'                 call    wrhex           ;write hex code
034B'  D1                       pop     de
034C'  C1                       pop     bc
034D'  10 EE                    djnz    wrhexf2
034F'  D5                       push    de
0350'  AF                       xor     a
0351'  99                       sbc     a,c
0352'  CD 0471'                 call    wrhex           ;write check sum
0355'  3E 0D                    ld      a,cr
0357'  CD 047B'                 call    wrhfl
035A'  3E 0A                    ld      a,lf
035C'  CD 047B'                 call    wrhfl           ;write cr,lf
035F'  D1                       pop     de
0360'  18 A4                    jr      wrhexf1
0362'                   wrhexf3:
0362'  21 0371'                 ld      hl,endrec
0365'                   wrhexf4:
0365'  7E                       ld      a,(hl)
0366'  B7                       or      a
0367'  28 16                    jr      z,wrhexf5
0369'  23                       inc     hl
036A'  E5                       push    hl
036B'  CD 047B'                 call    wrhfl           ;write end record
036E'  E1                       pop     hl
036F'  18 F4                    jr      wrhexf4
                        ;
```

```
0371'   3A 30 30 30     endrec: defb    ':00000001FF',cr,lf,0
0375'   30 30 30 30
0379'   31 46 46 0D
037D'   0A 00
                        ;
037F'                   wrhexf5:
037F'   3E 1A                   ld      a,eof
0381'   CD 047B'                call    wrhfl           ;eof write
0384'   3A 0000"                ld      a,(bufptr)
0387'   B7                      or      a
0388'   20 F5                   jr      nz,wrhexf5
038A'   11 005C                 ld      de,dfcb
038D'   0E 10                   ld      c,closef
038F'   CD 0005                 call    bdos            ;close file
0392'   3C                      inc     a
0393'   28 77                   jr      z,nocls
                        ;
0395'   21 03A3'                ld      hl,movtpa
0398'   ED 5B 000B"             ld      de,(objmov)
039C'   D5                      push    de
039D'   01 0022                 ld      bc,movtpae-movtpa
03A0'   ED B0                   ldir
03A2'   C9                      ret                     ;chain
                        ;
                        ;       move object to tpa & return to os
                        ;
03A3'                   movtpa::
03A3'   2A 0005"                ld      hl,(loc)
03A6'   ED 5B 0003"             ld      de,(prgorg)
03AA'   B7                      or      a
03AB'   ED 52                   sbc     hl,de
03AD'   CA 0000                 jp      z,boot
03B0'   44                      ld      b,h
03B1'   4D                      ld      c,l
03B2'   2A 0007"                ld      hl,(frebga)
03B5'   11 0100                 ld      de,tpa
03B8'   ED B0                   ldir                    ;move
03BA'   EB                      ex      de,hl
03BB'                   movtpal:
03BB'   7D                      ld      a,l
03BC'   B7                      or      a
03BD'   CA 0000                 jp      z,boot          ;return to os
03C0'   36 00                   ld      (hl),0
03C2'   23                      inc     hl
03C3'   18 F6                   jr      movtpal
03C5'                   movtpae:
                        ;
                        ;       error
                        ;
03C5'                   nofile:
03C5'   11 03CB'                ld      de,$+6
03C8'   C3 043B'                jp      error
03CB'   0D 0A 25 4E             defb    cr,lf,'%No object file$'
03CF'   6F 20 6F 62
03D3'   6A 65 63 74
03D7'   20 66 69 6C
03DB'   65 24
03DD'                   dirful:
03DD'   11 03E3'                ld      de,$+6
03E0'   C3 043B'                jp      error
03E3'   0D 0A 25 4E             defb    cr,lf,'%No directory space$'
03E7'   6F 20 64 69
03EB'   72 65 63 74
03EF'   6F 72 79 20
03F3'   73 70 61 63
03F7'   65 24
03F9'                   dskful:
03F9'   11 03FF'                ld      de,$+6
```

```
03FC'   C3 043B'                jp      error
03FF'   0D 0A 25 44             defb    cr,lf,'%Disk full$'
0403'   69 73 6B 20
0407'   66 75 6C 6C
040B'   24
040C'                   nocls:
040C'   11 0412'                ld      de,$+6
040F'   C3 043B'                jp      error
0412'   0D 0A 25 43             defb    cr,lf,'%Cannot close$'
0416'   61 6E 6E 6F
041A'   74 20 63 6C
041E'   6F 73 65 24
0422'                   badobj:
0422'   11 0428'                ld      de,$+6
0425'   C3 043B'                jp      error
0428'   0D 0A 25 42             defb    cr,lf,'%Bad object file$'
042C'   61 64 20 6F
0430'   62 6A 65 63
0434'   74 20 66 69
0438'   6C 65 24
                        ;
043B'                   error:
043B'   0E 09                   ld      c,printf
043D'   CD 0005                 call    bdos
0440'   C3 0000                 jp      boot            ;return to os
                        ;
                        ;       **  get object file  **
                        ;
0443'                   getobj::
0443'   3A 0000"                ld      a,(bufptr)
0446'   FE 80                   cp      reclen
0448'   38 17                   jr      c,getobj1
044A'   AF                      xor     a
044B'   32 0000"                ld      (bufptr),a
044E'   11 0080                 ld      de,dbuff
0451'   0E 1A                   ld      c,setdmf
0453'   CD 0005                 call    bdos            ;set dma address
0456'   11 005C                 ld      de,dfcb
0459'   0E 14                   ld      c,readf
045B'   CD 0005                 call    bdos            ;read object file
045E'   B7                      or      a
045F'   20 C1                   jr      nz,badobj
0461'                   getobj1:
0461'   2A 0000"                ld      hl,(bufptr)
0464'   26 00                   ld      h,0
0466'   11 0080                 ld      de,dbuff
0469'   19                      add     hl,de
046A'   7E                      ld      a,(hl)          ; a = object
046B'   21 0000"                ld      hl,bufptr
046E'   34                      inc     (hl)
046F'   6F                      ld      l,a
0470'   C9                      ret
                        ;
                        ;       **  write hex  **
                        ;
0471'                   wrhex:: ;               a : write data
0471'   CD 04F9'                call    binhex          ;convert to hex
0474'   D5                      push    de
0475'   7A                      ld      a,d
0476'   CD 047B'                call    wrhf1
0479'   D1                      pop     de
047A'   7B                      ld      a,e
                        ;
047B'                   wrhf1:
047B'   2A 0000"                ld      hl,(bufptr)
047E'   26 00                   ld      h,0
0480'   11 0080                 ld      de,dbuff
0483'   19                      add     hl,de
```

```
0484'   77              ld      (hl),a
0485'   21 0000"        ld      hl,bufptr
0488'   34              inc     (hl)
0489'   7E              ld      a,(hl)
048A'   FE 80           cp      reclen
048C'   D8              ret     c
048D'   36 00           ld      (hl),0
048F'   11 0080         ld      de,dbuff
0492'   0E 1A           ld      c,setdmf
0494'   CD 0005         call    bdos            ;set dma address
0497'   11 005C         ld      de,dfcb
049A'   0E 15           ld      c,writef
049C'   CD 0005         call    bdos            ;write file
049F'   B7              or      a
04A0'   C2 03F9'        jp      nz,dskful
04A3'   C9              ret
                ;
                ;       ** print decimal **
                ;
04A4'           prdec:: ;       hl: print data
04A4'   16 00           ld      d,0
04A6'   01 2710         ld      bc,10000
04A9'   CD 04C7'        call    prdecs
04AC'   01 03E8         ld      bc,1000
04AF'   CD 04C7'        call    prdecs
04B2'   01 0064         ld      bc,100
04B5'   CD 04C7'        call    prdecs
04B8'   01 000A         ld      bc,10
04BB'   CD 04C7'        call    prdecs
04BE'   7D              ld      a,l
04BF'   F6 30           or      '0'
04C1'   5F              ld      e,a
04C2'   0E 02           ld      c,putchrf
04C4'   C3 0005         jp      bdos
                ;
04C7'           prdecs:
04C7'   1E 30           ld      e,'0'
04C9'           prdecs1:
04C9'   B7              or      a
04CA'   ED 42           sbc     hl,bc
04CC'   1C              inc     e
04CD'   30 FA           jr      nc,prdecs1
04CF'   09              add     hl,bc
04D0'   1D              dec     e
04D1'   7B              ld      a,e
04D2'   FE 30           cp      '0'
04D4'   20 03           jr      nz,prdecs2
04D6'   15              dec     d
04D7'   14              inc     d
04D8'   C8              ret     z
04D9'           prdecs2:
04D9'   14              inc     d
04DA'   E5              push    hl
04DB'   D5              push    de
04DC'   0E 02           ld      c,putchrf
04DE'   CD 0005         call    bdos
04E1'   D1              pop     de
04E2'   E1              pop     hl
04E3'   C9              ret
                ;
                ;       ** print hex **
                ;
04E4'           prhex4:: ;      hl:print data
04E4'   E5              push    hl
04E5'   7C              ld      a;h
04E6'   CD 04EB'        call    prhex2
04E9'   E1              pop     hl
04EA'   7D              ld      a,l
```

```
04EB'                   prhex2:
04EB'   CD 04F9'                call    binhex          ;convert to hex
04EE'   D5                      push    de
04EF'   5A                      ld      e,d
04F0'   CD 04F4'                call    prhex1
04F3'   D1                      pop     de
04F4'                   prhex1:
04F4'   0E 02                   ld      c,putchrf
04F6'   C3 0005                 jp      bdos
                        ;
                        ;       **  convert to hex  **
                        ;
04F9'                   binhex:: ;      a : 8 bit binary
04F9'   F5                      push    af
04FA'   0F                      rrca
04FB'   0F                      rrca
04FC'   0F                      rrca
04FD'   0F                      rrca
04FE'   CD 0503'                call    binhex1
0501'   53                      ld      d,e
0502'   F1                      pop     af
0503'                   binhex1:
0503'   E6 0F                   and     0fh
0505'   C6 30                   add     a,'0'
0507'   FE 3A                   cp      '9'+1
0509'   38 02                   jr      c,$+4
050B'   C6 07                     add   a,'A'-'9'-1
050D'   5F                      ld      e,a
050E'   C9                      ret
                        ;
                        ;       **  make symbol file  **
                        ;
050F'                   mksyfl::
050F'   21 005C                 ld      hl,dfcb
0512'   11 000E"                ld      de,sfcb
0515'   D5                      push    de
0516'   01 0009                 ld      bc,9
0519'   ED B0                   ldir                    ;set symbol file fcb
051B'   EB                      ex      de,hl
051C'   36 53                   ld      (hl),'S'
051E'   23                      inc     hl
051F'   36 59                   ld      (hl),'Y'
0521'   23                      inc     hl
0522'   36 4D                   ld      (hl),'M'
0524'   23                      inc     hl
0525'   AF                      xor     a
0526'   77                      ld      (hl),a
0527'   23                      inc     hl
0528'   77                      ld      (hl),a
0529'   23                      inc     hl
052A'   77                      ld      (hl),a
052B'   32 002E"                ld      (sfcb+32),a     ;clear sfcb.ex, s1. s2. cr
052E'   D1                      pop     de
052F'   D5                      push    de
0530'   0E 13                   ld      c,deletf
0532'   CD 0005                 call    bdos            ;delete old sym file
0535'   D1                      pop     de
0536'   0E 16                   ld      c,makef
0538'   CD 0005                 call    bdos            ;make new sym file
053B'   3C                      inc     a
053C'   CA 03DD'                jp      z,dirful
053F'   21 0000                 ld      hl,0
0542'   22 0030"                ld      (sbptr),hl
0545'   3E FF                   ld      a,0ffh
0547'   32 000D"                ld      (symotf),a
054A'   C9                      ret
                        ;
                        ;       **  write symbol file  **
```

```
                        ;
054B'                   wrsymf:: ;      a : write character
054B'   FE 61                   cp      'a'
054D'   38 06                   jr      c,wrsymf0
054F'   FE 7B                   cp      'z'+1
0551'   30 02                   jr      nc,wrsymf0
0553'   D6 20                   sub     20h             ;translate to upper case
0555'                   wrsymf0:
0555'   2A 0030"                ld      hl,(sbptr)
0558'   E5                      push    hl
0559'   11 0032"                ld      de,symbuf
055C'   19                      add     hl,de
055D'   77                      ld      (hl),a          ;buffer store
055E'   E1                      pop     hl
055F'   23                      inc     hl
0560'   22 0030"                ld      (sbptr),hl
0563'   11 0200                 ld      de,sbsiz
0566'   B7                      or      a
0567'   ED 52                   sbc     hl,de           ; sbptr >= sbsiz ?
0569'   D8                      ret     c
                        ;
056A'                   wrsymf1:
056A'   2A 0030"                ld      hl,(sbptr)
056D'   11 0032"                ld      de,symbuf
0570'                   wrsymf2:
0570'   E5                      push    hl
0571'   D5                      push    de
0572'   0E 1A                   ld      c,setdmf
0574'   CD 0005                 call    bdos            ;set dma address
0577'   11 000E"                ld      de,sfcb
057A'   0E 15                   ld      c,writef
057C'   CD 0005                 call    bdos            ;write file
057F'   B7                      or      a
0580'   C2 03F9'                jp      nz,dskful
0583'   01 0080                 ld      bc,reclen
0586'   E1                      pop     hl
0587'   09                      add     hl,bc
0588'   EB                      ex      de,hl
0589'   E1                      pop     hl
058A'   B7                      or      a
058B'   ED 42                   sbc     hl,bc
058D'   20 E1                   jr      nz,wrsymf2
058F'   22 0030"                ld      (sbptr),hl
0592'   C9                      ret
                        ;
                        ;       **  close symbol file  **
                        ;
0593'                   clsymf::
0593'   3A 002F"                ld      a,(symhcu)
0596'   B7                      or      a
0597'   28 0A                   jr      z,clsymf1
0599'   3E 0D                   ld      a,cr
059B'   CD 054B'                call    wrsymf
059E'   3E 0A                   ld      a,lf
05A0'   CD 054B'                call    wrsymf
05A3'                   clsymf1:
05A3'   ED 5B 0030"             ld      de,(sbptr)
05A7'                   clsymf2:
05A7'   21 0032"                ld      hl,symbuf
05AA'   19                      add     hl,de
05AB'   36 1A                   ld      (hl),eof
05AD'   13                      inc     de
05AE'   7B                      ld      a,e
05AF'   E6 7F                   and     reclen-1
05B1'   20 F4                   jr      nz,clsymf2
05B3'   ED 53 0030"             ld      (sbptr),de
05B7'   CD 056A'                call    wrsymf1         ;disk write
05BA'   11 000E"                ld      de,sfcb
```

```
05BD'   0E 10              ld      c,closef
05BF'   CD 0005            call    bdos
05C2'   3C                 inc     a
05C3'   CA 040C'           jp      z,nocls
05C6'   C9                 ret
                   ;
                   ;
05C7'              $memry::defs    2       ;free area address ( LINK-80 set )


                   ;-----------------------------------------------------
                   ;-                  work area                         -
                   ;-----------------------------------------------------

05C9'                      dseg

0000"              bufptr::defs    1       ;buffer pointer
0001"              offset::defs    2       ;address offset
0003"              prgorg::defs    2       ;program origin
0005"              loc::   defs    2       ;location counter
0007"              frebga::defs    2       ;free area beginning address
0009"              freeda::defs    2       ;free area ending address
000B"              objmov::defs    2       ;object tpa move routine entry address
000D"              symotf::defs    1       ;symbol file output flag
000E"              sfcb::  defs    33      ;symbol file fcb
002F"              symhcu::defs    1       ;symbol horizontal out count
0030"              sbptr:: defs    2       ;sym file output buffer pointer
0032"              symbuf::defs    reclen*4        ;symbol file output buffer
0200               sbsiz   equ     $-symbuf        ;buffer size

                           end     start
```

第3章

PC-8801バージョンのStellarコンパイラ

PC-8801/mk IIのN88-BASIC上で動作するStellarコンパイラの使用法，ならびに各種ライブラリについての説明をします。なお，ディスクの使用を前提にしているので，カセットで使用する場合は操作が異なったり，使えないライブラリがあるので注意してください。

3-1 コンパイラの使用法

N88－BASIC上でStellarコンパイラを使うときは，次の操作を行います。

①電源をONするか，リセット・スイッチを押す。

② How many files（0～15）？と表示されたら0～2のいずれかの数字を入力する。

③ディスクからStellarコンパイラをロードする。たとえばコンパイラがファイル名stelar.b5でドライブ1にあるなら，次のコマンドを入力する。

```
CLEAR , &HB4FF⏎
BLOAD "stelar. b5", R⏎
```

以上の操作により，N88－BASICにStellarコンパイラのための次の五つのコマンドが追加され，いつでも使用できる状態になります。

CMD COMP…………コンパイルし，オブジェクト・プログラムをVRAM上へ出力する。CMD COMP "P"とすると，コンパイル中のメッセージ（CP／Mバージョンと同じ内容）をプリンタにも印字する。

CMD LOAD…………VRAM上のオブジェクト・プログラムをメイン・メモリへロードする。CMD LOAD "P"とすると，ロード中のメッセージ（CP／MバージョンのCONVOBJと同じ内容)をプリンタにも印字する。ロードされたオブジェクト・プログラムは，CMD RUN，USR関数，CALL文によって実行できる。

CMD RUN…………ロードしたオブジェクト・プログラムを実行する。実行開始アドレスはCMD LOADによって設定される。デバッグ・モードでコンパイルされたプログラムは必ずこのCMD RUNで実行しなければならない。また，CMD RUN &Hhhhhとするとhhhh番地から実行

することができる。これは引数のないUSR関数，CALL文と思えばよい。一度CMD RUN &Hhhhhを実行すると，次のCMD RUNはhhhh番地から実行されるようになる。

CMD CONT …………％breakなどで中断したプログラムの実行を再開する。

CMD LIST 〝文字列〟………ストリング・サーチ。Stellarのソース・プログラムから指定された文字列を含む行を探し，該当する行をすべてディスプレイに表示する。CMD LLIST 〝文字列〟とすると，表示する内容をそのままプリンタにも印字するようになる。

これらのコマンドは実行中，**表3-1**のキーを押せば一時停止，中断ができます。

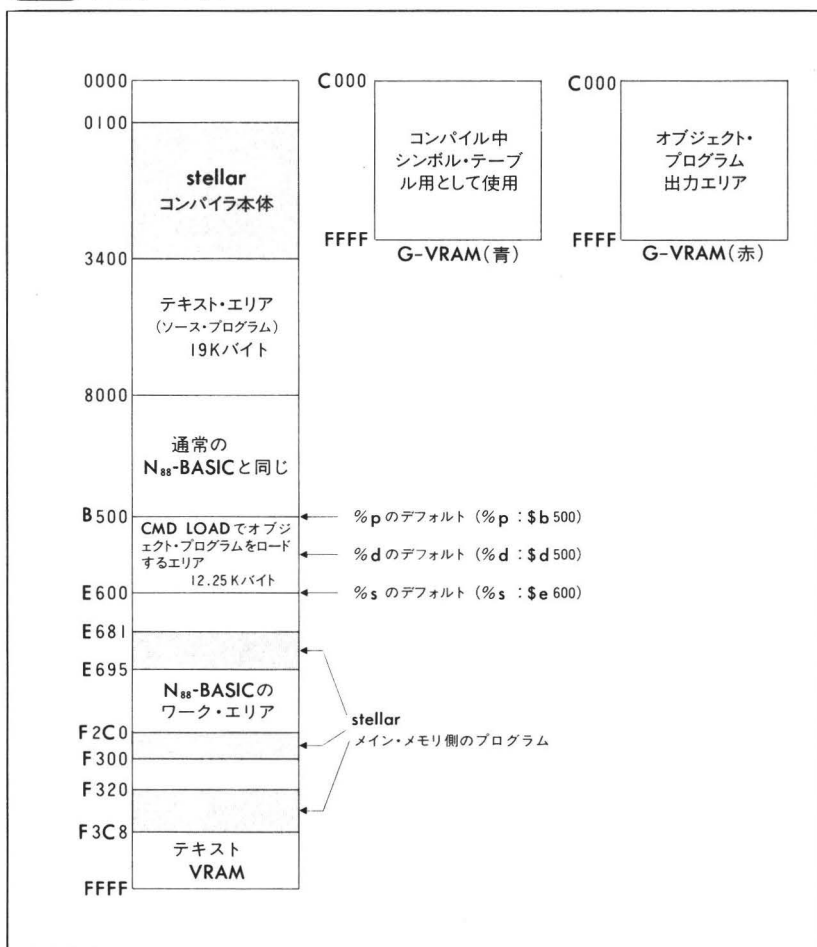
図3-1はStellarコンパイラ使用中のＰＣ-8801のメモリの状態を表したものです。

表3-1 一時停止，中断の仕方

コマンド	一時停止	一時停止の解除	中　断
CMD COMP	な し	な し	な し
CMD LOAD	ESCキー	スペース・キー	な し
CMD LIST	ESCキー	スペース・キー	一時停止中に STOPキー
CMD LLIST	ESCキー	スペース・キー	一時停止中に STOPキー
CMD RUN※1	ESCキー	スペース・キー	STOPキー

※1 CMD RUNはデバック・モードで％tronのときのみ有効

図3-1 メモリ・マップ

0000
0100
stellar
コンパイラ本体
3400
テキスト・エリア
（ソース・プログラム）
19Kバイト
8000
通常の
N88-BASICと同じ
B500
CMD LOADでオブジェクト・プログラムをロードするエリア
12.25Kバイト
E600
E681
E695
N88-BASICの
ワーク・エリア
F2C0
F300
F320
F3C8
テキスト
VRAM
FFFF
C000
コンパイル中
シンボル・テーブル用として使用
FFFF
G-VRAM（青）
C000
オブジェクト・
プログラム
出力エリア
FFFF
G-VRAM（赤）
%pのデフォルト（%p：$b500）
%dのデフォルト（%d：$d500）
%sのデフォルト（%s：$e600）
stellar
メイン・メモリ側のプログラム

3-2 ソース・プログラムのつくりかた

Stellarのソース・プログラムはN88－BASICの注釈行（引用符'による）でつくります。その形式は次の通りです。

```
行番号 ' 1行分のソース・プログラム
```

ソース・プログラム中にはREMによる注釈や一般のBASIC文を書いてはなりません。

例.

```
100 'prog tst ( ) ;
520 '  a :=a/3 ;
```

3-3 使用上の注意

Stellarコンパイラを N88－BASIC上で動作させているときは，次のような点に注意してください。

①普通のBASICのプログラムを走らせるときは，実行を始める行番号をつけて

RUN　行番号

としてください。

②裏RAMの0001番地～0002番地にゼロ以外の値を入れてRUNすると暴走することがあります。

③Stellarコンパイラをロードするとき，0000番地からのBASICのテキストを3400番地以降に転送します。このとき0000番地以降にBASICのテキスト以外のプログラムが入っていると暴走することがあります（このようなときは一度リセットしてからロードする)。

④③の転送の際，BASICのテキストが7FFF番地を超えるとOut of memoryのエラーを出します。この場合，N88－BASICにはStellarコンパイラのためのコマンドは追加されません。

⑤PC-8801 mkⅡに元来あるCMD使用の拡張コマンドは使えなくなります。

⑥コンパイラはオブジェクト・プログラムを一度VRAM（赤）へ出力するので，コンパイル後CLS 3などの命令でグラフィック画面を消すとオブジェクト・プログラムも消えてしまいます。

⑦CMD COMPとCMD LOADを実行したときは，SCREEN, 3を実行したのと同じ状態になります。

⑧AUTOコマンド実行時に行番号の後に引用符（'）が出ますが，これをやめるときはＥＤ１８Ｈ番地にＣ９Ｈを書き込んでください。

⑨デバッグ・モードにおいて（%breakや実行中のSTOPキーにより）中断したときは，モニタで中断時のレジスタの値を見たり，レジスタの値を変更してCMD CONTで実行を再開することができます。

レジスタの読み出しはZRで，書き込みはZWで行います。読み出し，書き込みができるのは，AF，BC，DE，HL，IX，IY，PC，SPの各レジスタです。

3-4 ライブラリの使いかた

ＰＣ-8801用のライブラリには，基本ライブラリ，コンソール入出力ライブラリ，算術ライブラリ，ファイル入出力ライブラリ，物理ディスク入出力ライブラリの五つがあります。これらのライブラリを使うときは，MERGEコマンドで自分で作成したプログラムの一部として組み込みます。

〔1〕基本ライブラリ

基本ライブラリには次の六つの関数があります。

（i） **console** (〈スクロール開始行〉，〈スクロール行数〉，〈ファンクション・キー表示スイッチ〉，〈カラー／白黒スイッチ〉)

（ii） **width** (〈桁数〉，〈行数〉)

（iii） **screen** (〈画面モード〉，〈画面スイッチ〉)

（iv） **color** (〈ファンクション・コード〉)

（v） **cls** (〈機能〉)

（vi） **locate** (〈X座標〉，〈Y座標〉，〈カーソル・スイッチ〉)

注）〈　〉で囲まれた項目はパラメータの内容を示す。また，（iii）（iv）では残りのパラメータは指定できない。（v）ではどの画面モードのときでもＢＲＧの三画面をすべてクリアする。

機能はN88－BASICのコマンドとほぼ同じです。変更したくないパラメータは－１を指定してください。

●リスト3-1　基本ライブラリ

```
10000 ' /* basic                   */
10010 ' /*      1.console          */
10020 ' /*      2.width            */
10030 ' /*      3.screen           */
10040 ' /*      4.color            */
10050 ' /*      5.cls              */
10060 ' /*      6.locate           */
10070 ' /*                                  */
10080 ' /*         written by K.Isizuka     */
10090 ' /*                 09/05/84         */
10100 ' /*                                  */
10110 ' console(s,n,F,C:#,#);
10120 '   var CNSDFG at ($e6b8),LINEND at ($e6b1);
10130 '   var SCRLL1 at ($e6b2),SCRLL2 at ($e6b3);
10140 '   var CMODE  at ($e6b9);
10150 '   {
10160 '     SCRLL1:=?(s<=25;s+1,?(not s;25,SCRLL1));
10170 '     SCRLL2:=?(n<=25;?(n;n+s,1),?(not n;25,SCRLL2));
10180 '     if not F
10190 '       then{
10200 '         if F
10210 '           then CNSDFG:=$ff;
10220 '           else {inline $cd,#$4021;CNSDFG:=0;}
10230 '       }
10240 '     C:=?(not C;C,CMODE);
10250 '     inline
10260 '           $3a,#.C,       /* ld   a,(.C)  */
10270 '           $cd,#$70d1;    /* call 70d1h   */
10280 '   }
10290 ' width(h,v;#,#);
10300 '   var LINCNT at ($ef88),LINWDT at ($ef89);
10310 '   {
10320 '     if not h then;else h:=LINWDT;
10330 '     if not v then;else v:=LINCNT;
10340 '     if (h=40 or h=80) and (v=25 or v=20) then
10350 '       inline
10360 '           $2a,#.h,       /* ld   hl,(.h) */
10370 '           $45,           /* ld   b,l     */
10380 '           $4c,           /* ld   c,h     */
10390 '           $cd,#$6f6b;    /* call 6f6b    */
10400 '   }
10410 'screen(c,f;#,#);
10420 '  var HIRESL at ($e6a6),PORT31 at ($e6c2);
10430 '  var PORT40 at ($e6c1);
10440 '  {
10450 '    if inc(c)
10460 '      then{
10470 '        if dec(c)
10480 '          then{
10490 '            if dec(c)
10500 '              then{
10510 '                if dec(c)
10520 '                  then;
10530 '                  else{ HIRESL:=1;port[$31]:=PORT31:=PORT31 and$ee;}
10540 '              }
10550 '              else{ HIRESL:=0;port[$31]:=PORT31:=(PORT31 or$01)and$ef;}
10560 '          }
10570 '          else{ HIRESL:=0;port[$31]:=PORT31:=PORT31 or$11;}
10580 '      }
10590 '    if not f then
10600 '      {
10610 '        if f and $01
10620 '          then port[$40]:=PORT40:=PORT40 or $10;
10630 '          else port[$40]:=PORT40:=PORT40 and $ef;
10640 '        if f and $02
10650 '          then port[$31]:=PORT31:=PORT31 and $f7;
10660 '          else port[$31]:=PORT31:=PORT31 or $08;
```

```
10670 '        }
10680 '    }
10690 ' color(function_code;#,#);
10700 '    var NULATR at ($e6b4),NULCHR at (.NULATR+1);
10710 '    var F_LTRL at (.NULATR+2),CMODE at ($e6b9);
10720 '    {
10730 '      if CMODE
10740 '        then NULATR:=(function_code<<5) or $08;
10750 '        else NULATR:=function_code and $07;
10760 '    }
10770 ' cls(n;#,#);
10780 '    {
10790 '      if n and $01 then inline $cd,#$5f0e;
10800 '      if n and $02 then
10810 '          inline
10820 '            $f3,            /* di            */
10830 '            $3a,#$e6c1,     /* ld   a,(PORT40)*/
10840 '            $f6,$10,        /* or   10h      */
10850 '            $d3,$40,        /* out  (40h),a */
10860 '            $3e,$5c,        /* ld   a,5ch    */
10870 '            $4f,            /* lp:  ld   c,a      */
10880 '            $ed,$79,        /* out  (c),a    */
10890 '            $01,#$3e7f,     /* ld   bc,3e7fh*/
10900 '            $11,#$c001,     /* ld   de,c001h*/
10910 '            $21,#$c000,     /* ld   hl,c000h*/
10920 '            $36,$00,        /* ld   (hl),0  */
10930 '            $ed,$b0,        /* ldir          */
10940 '            $3c,            /* inc  a        */
10950 '            $fe,$5f,        /* cp   5fh      */
10960 '            $20,$eb,        /* jr   nz,lp    */
10970 '            $d3,$5f,        /* out  (5fh),a */
10980 '            $3a,#$e6c1,     /* ld   a,(e6c1h)*/
10990 '            $d3,$40,        /* out  (40h),a */
11000 '            $fb;            /* ei            */
11010 '   }
11020 'locate(X,Y,F;#,#);
11030 '   var CSRY at ($ef86),CSRX at (.CSRY+1);
11040 '   var LINCNT at (.CSRY+2),LINWDT at (.CSRY+3);
11050 '   var CURFG at ($e6a7);
11060 '   {
11070 '     if not Y then CSRY:=?(LINCNT>Y;Y+1,LINCNT);
11080 '     if not X then CSRX:=?(LINWDT>X;X+1,LINWDT);
11090 '     if not F then CURFG:=?(F;-1,0);
11100 '   }
```

〔２〕コンソール入出力ライブラリ

コンソール（ＣＲＴ，キーボード）との入出力やプリンタへの出力のための関数です。

（ｉ）　**print_str**（；〈文字列の開始アドレス〉）
…ＮＵＬコード(ゼロ)で終わる文字列を出力する。

（ii）　**print_num**（〈下位バイト〉，〈上位バイト〉）
…符号なしの２バイトの数を10進で出力する。

（iii）　**print_hex**（〈数〉）
…数を16進で出力する。

（iv）　**print_using**（〈出力する桁数〉，〈下位バイト〉，〈上位バイト〉）
…符号つきの２バイトの数を10進で出力する。指定された桁数で，右づめ出力される。

（ｖ）　**print_chr**（〈数〉）
…数をそのまま文字として出力する。

（vi）　**input_str**（〈文字列の長さ〉；〈変数の先頭アドレス〉）
…リターン・キーが押されるまでに入力した文字列を〈変数の先頭アドレス〉から代入し，文字列の最後にＮＵＬコードを入れる。入力した文字列が〈文字列の長さ〉よりも長いときは，文字列の長さ－１文字分だけを代入する。入力された実際の文字列の長さは関数の値として返される。STOP キーが入力されたときは，代入せずにキャリーフラグを立てて戻る。

（vii）　**input_num**（〈type〉；〈変数のアドレス〉）
…〈type〉が１のときは，キーボードから入力された値（10進数）を１バイトの値に変換し〈変数のアドレス〉に代入する。〈type〉が１以外のときは，キーボードから入力された値（符号つきの10進数）を符号つきの２バイトの値に変換し，〈変数のアドレス〉に下位バイトを，次のアドレスに上位バイトを代入する。STOP キーが

入力された場合には1を，入力した値が指定した <type> と一致しない場合は2を関数の値として返す。どちらの場合もキャリーフラグが立ち，入力が正しく行われなかったことを示す。

(viii) **input _ chr**（ ）

…キーボードから入力された文字を関数の値として返す（入力待ちあり)。

(ix) **inkey**（ ）

…キーボードから入力された文字を関数の値として返す（入力待ちなし，入力がないときはゼロフラグが立つ)。

(x) **crlf**（<回数>)

… <回数> だけ復改を出力する。

(xi) **lpt _ sw**（<スイッチ>)

… <スイッチ>＝0なら出力先をＣＲＴにする。
<スイッチ>＝ff Hなら現在の出力先を関数の値として返す（0：ＣＲＴ，ff H：ＬＰＴ)。
<スイッチ>＝その他なら出力先をＬＰＴにする。

(xii) **sense**（ ）

…キー入力に応じた働きをする。
^Ｓ：一時停止。
^Ｏ：ＣＲＴへ出力しない。
^Ｃ：実行停止。

●リスト3-2　　コンソール入出力ライブラリ

```
20000 '/*I/O console            */
20010 '/*       1.print_str     */
20020 '/*       2.print_num     */
20030 '/*       3.print_hex     */
20040 '/*       4.print_using   */
20050 '/*       5.print_chr     */
20060 '/*       6.input_str     */
20070 '/*       7.input_num     */
20080 '/*       8.input_chr     */
20090 '/*       9.inkey         */
20100 '/*       10.crlf         */
20110 '/*       11.lpt_sw       */
20120 '/*       12.sense        */
20130 '/*                                   */
20140 '/*         written by K.Isizuka      */
20150 '/*                  09/05/84         */
20160 '/*                                   */
20170 ' var  PRTFLG at($e64c);
20180 ' data _lpt:00;
20190 ' print_str(len;ix,#);
20200 '   {
20210 '     PRTFLG:=_lpt;
20220 '     inline
20230 '           $dd,$e5,       /* push ix      */
20240 '           $e1,           /* pop  hl      */
20250 '           $7e,           /*l1:ld   a,(hl) */
20260 '           $b7,           /* or   a       */
20270 '           $28,$04,       /* jp   z,$+6   */
20280 '           $df,           /* rst  18h     */
20290 '           $23,           /* inc  hl      */
20300 '           $18,$f8;       /* jr   l1      */
20310 '     PRTFLG:=0;
20320 '   }
20330 ' print_num(num_l,num_h;#,#);
20340 '   {
20350 '     PRTFLG:=_lpt;
20360 '     inline
20370 '           $2a,#.num_l,   /* ld   hl,(.num_l)      */
20380 '           $cd,#$28c2;    /* call 28c2h   */
20390 '     PRTFLG:=0;
20400 '   }
20410 ' print_hex(num;#,#);
20420 '   data _lbl_:
20430 '           $c6,$30,       /* add  30h     */
20440 '           $fe,':',       /* cp   ':'     */
20450 '           $38,$02,       /* jr   c,02    */
20460 '           $c6,$07,       /* add  07h     */
20470 '           $df,           /* rst  18h     */
20480 '           $c9;           /* ret          */
20490 '   {
20500 '     PRTFLG:=_lpt;
20510 '     inline
20520 '           $3a,#.num,     /* ld   a,(.num)*/
20530 '           $e6,$f0,       /* and  f0h     */
20540 '           $07,           /* rlca         */
20550 '           $07,           /* rlca         */
20560 '           $07,           /* rlca         */
20570 '           $07,           /* rlca         */
20580 '           $cd,#._lbl_,   /* call ._lbl_  */
20590 '           $3a,#.num,     /* ld   a,(.num)*/
20600 '           $e6,$0f,       /* and  0fh     */
20610 '           $cd,#._lbl_;   /* call ._lbl_  */
20620 '     PRTFLG:=0;
20630 '   }
20640 ' print_using(len,low_num,high_num;#,#);
20650 '   {
20660 '     PRTFLG:=_lpt;
```

```
20670 '     inline
20680 '           $2a,#.low_num,/* ld   hl,(.low_num)    */
20690 '           $cd,$fd,$21,  /* call 21fdh   */
20700 '           $3a,#.len,    /* ld   a,(.len)*/
20710 '           $47,          /* ld   b,a     */
20720 '           $3e,$80,      /* ld   a,80h   */
20730 '           $0e,$00,      /* ld   c,00    */
20740 '           $cd,$d1,$28,  /* call 28d1h   */
20750 '           $cd,#$5550;   /* call 5550h   */
20760 '     PRTFLG:=0;
20770 '   }
20780 ' print_chr(chr;#,#);
20790 '   {
20800 '     PRTFLG:=_lpt;
20810 '     inline
20820 '           $3a,#.chr,    /* ld   a,(.chr)*/
20830 '           $df;          /* rst  18h     */
20840 '     PRTFLG:=0;
20850 '   }
20860 ' input_str(len;ix,#);
20870 '   {
20880 '     inline
20890 '           $c5,          /* push bc      */
20900 '           $cd,#$5fc8,   /* call 5fc8h   */
20910 '           $38,$19,      /* jr   c,lbl   */
20920 '           $3a,#.len,    /* ld   a,(.len)*/
20930 '           $47,          /* ld   b,a     */
20940 '           $0e,$00,      /* ld   c,00    */
20950 '           $05,          /* dec  b       */
20960 '           $dd,$e5,      /* push ix      */
20970 '           $d1,          /* pop  de      */
20980 '           $eb,          /* ex   de,hl   */
20990 '           $13,          /*lp:inc  de      */
21000 '           $1a,          /* ld   a,(de)  */
21010 '           $77,          /* ld   (hl),a  */
21020 '           $b7,          /* or   a       */
21030 '           $79,          /* ld   a,c     */
21040 '           $28,$07,      /* jr   z,lbl   */
21050 '           $23,          /* inc  hl      */
21060 '           $0c,          /* inc  c       */
21070 '           $10,$f5,      /* djnz .lp     */
21080 '           $36,$00,      /* ld   (hl),0  */
21090 '           $79,          /* ld   a,c     */
21100 '           $c1;          /*lbl:pop  bc      */
21110 '   }
21120 ' input_num(type;ix,#);
21130 '   data _lbl_:
21140 '           $cd,#$5fc8,   /* call 5fc8h   */
21150 '           $3e,$01,      /* ld   a,01h   */
21160 '           $d8,          /* ret  c       */
21170 '           $23,          /* inc  hl      */
21180 '           $cd,#$26bc,   /* call 26bch   */
21190 '           $f7,          /* rst  30h     */
21200 '           $3e,$02,      /* ld   a,02h   */
21210 '           $f0,          /* ret  p       */
21220 '           $2a,#$ec41,   /* ld   hl,(ec41h)       */
21230 '           $3a,#.type,   /* ld   a,(.type)        */
21240 '           $dd,$75,00,   /* ld   (ix),l  */
21250 '           $3d,          /* dec  a       */
21260 '           $28,$05,      /* jr   z,l1    */
21270 '           $dd,$74,01,   /* ld   (ix+1),h*/
21280 '           $af,          /* xor  a       */
21290 '           $c9,          /* ret          */
21300 '           $7c,          /*l1:ld   a,h     */
21310 '           $b7,          /* or   a       */
21320 '           $c8,          /* ret  z       */
21330 '           $3e,$02,      /* ld   a,02    */
21340 '           $37,          /* scf          */
```

```
21350 '          $c9;          /* ret          */
21360 '   {
21370 '     inline $cd,#._lbl_; /* call ._lbl_  */
21380 '   }
21390 ' input_chr(;#,#);
21400 '   {
21410 '     inline
21420 '          $cd,#$3583;   /* call 3583h   */
21430 '   }
21440 ' inkey(;#,#);
21450 '   {
21460 '     inline
21470 '          $cd,#$35ce;   /* call 35ceh   */
21480 '   }
21490 ' crlf(num;#,#);
21500 '   var i at(_work+30);
21510 '   cons CR:=13,LF:=10;
21520 '   {
21530 '     for i:=1 to num {print_chr(CR);print_chr(LF);}
21540 '   }
21550 ' lpt_sw(sw;#,#);
21560 '   var LPTFLG at(._lpt);
21570 '   {
21580 '     if not(sw)
21590 '       then LPTFLG:=?(sw;-1,0);
21600 '       else LPTFLG;
21610 '   }
21620 ' sense(;#,#);
21630 '   var a at(_work+30),CNTOFL at($e652);
21640 '   {
21650 '     if zero(a:=inkey()) then return;
21660 '     if a=$f then {
21670 '       print_chr('^');print_chr('O');crlf(1);
21680 '       CNTOFL:=not CNTOFL;
21690 '       print_chr('^');print_chr('O');crlf(1);
21700 '     }
21710 '     if a=$3 then {print_chr(7);stop;}
21720 '     if a=$13 then {if input_chr()=$03 then{print_chr(7);stop;}}
21730 '   }
```

〔3〕算術ライブラリ

算術ライブラリには次の13個の関数があります。

（i） **add＿w**(a＿l，a＿h，b＿l，b＿h；〈アドレス〉)

…ワード長の加算。

（ii） **sub＿w**(a＿l，a＿h，b＿l，b＿h；〈アドレス〉)

…ワード長の減算。

（iii） **mul**（a，b）

…バイト長の乗算。結果はa＊bの下位バイト。

（iv） **mulov**（ ）

…mul関数の乗算結果の上位バイトの取り出し。

（v） **mul＿w**(a＿l，a＿h，b＿l，b＿h；〈アドレス〉)

…ワード長の乗算。

（vi） **mulov＿w**（；〈アドレス〉)

…mul＿w関数の乗算結果の上位バイトの取り出し。

（vii） **div**（a，b）

…バイト長の除算。結果はa／bの値。

（viii） **mod**（ ）

… ｄｉｖ関数の除算の余りを取り出す。

（ix） **div＿w**(a＿l，a＿h，b＿l，b＿h；〈アドレス〉)

…ワード長の除算。

（x） **mod＿w**（；〈アドレス〉)

…div＿w関数の除算の余りを取り出す。

（xi） **array**（x，y；〈アドレス〉)

…2次元配列の配列要素のアドレスを求める。

array（x，y，z；〈アドレス〉)

…3次元配列の配列要素のアドレスを求める。

（xii） **load＿array**（x，y；〈アドレス〉)

…2次元配列の配列要素の参照。

load_array（x，y，z；〈アドレス〉）
…3次元配列の配列要素の参照。

（xiii）　**store_array**（a，x，y；〈アドレス〉）
…2次元配列の配列要素へaの値を代入。
store_array（a，x，y，z；〈アドレス〉）
…3次元配列の配列要素へaの値を代入。

このうちadd_w，sub_w，mul_w，div_wはパラメータの渡しかたが同じです。つまりa_lとa_h，b_lとb_hをそれぞれ一つの2バイトの数（_lが下位，_hが上位）と見なし，符号なし2進整数として計算して結果の下位バイトを〈アドレス〉へ，上位バイトを〈アドレス〉＋1へ代入します。このとき，add_w，sub_wは計算結果に桁上がりや桁下がり（ボロー）が発生していればキャリーフラグを立てます。

mul_wの計算結果は一般的に4バイト長になりますが，この関数では下位2バイトだけを結果として返します。残る上位2バイトはmulov_wを呼べば求められます。求めた値の下位バイトは〈アドレス〉へ，上位バイトは〈アドレス〉＋1へ代入します。div_wで発生した余りをmod_wで求めることができます。求めた余りの下位バイトは〈アドレス〉へ，上位バイトは〈アドレス〉＋1へ代入します。

mul，divは1バイト長の計算（符号なし2進定数）でパラメータの渡しかたは同じです。計算結果は関数の値として返されます。mulovはmulの計算結果の上位バイトを求めるもので，これも結果は関数の値として返されます。modはdivの除算で発生した余りを求めるもので，結果は関数の値として返されます。

array，load_array，store_arrayは，Stellarでサポートしていない2次元，3次元の配列を扱うための関数です。2次元，3次元の配列は次のように宣言しなければなりません。

●2次元配列の宣言

data〈配列名〉:〈ワークの先頭アドレス〉,〈xの長さ〉,〈yの長さ〉, 1;

●3次元配列の宣言

data〈配列名〉:〈ワークの先頭アドレス〉,〈xの長さ〉,〈yの長さ〉,〈zの長さ〉;

ここで〈ワーク〉とは,配列要素を記憶するための領域で,大きさは〈xの長さ〉*〈yの長さ〉*〈zの長さ〉以上でなければなりません。

array はx,yあるいはx,y,zで示された配列要素(〈配列名〉[x, y]や〈配列名〉[x, y, z])のアドレスを求める関数で,結果のアドレスはインデックス・レジスタIXに設定されてきます。

load _ array は配列要素(同上)を参照する関数で,参照した値を関数値として返します。

store _ array は配列要素(同上)へaの値を代入する関数です。

●リスト3-3　算術ライブラリ

```
30000 '/* arithmetic            */
30010 '/*       1.add_w         */
30020 '/*       2.sub_w         */
30030 '/*       3.mul           */
30040 '/*       4.mulov         */
30050 '/*       5.mul_w         */
30060 '/*       6.mulov_w       */
30070 '/*       7.div           */
30080 '/*       8.mod           */
30090 '/*       9.div_w         */
30100 '/*      10.mod_w         */
30110 '/*      11.array         */
30120 '/*      12.load_array    */
30130 '/*      13.store_array   */
30140 '/*                                 */
30150 '/*         written by K.Isizuka    */
30160 '/*                  09/05/84       */
30170 '/*                                 */
30180 '
30190 'add_w(a_l,a_h,b_l,b_h;ix,#);
30200 '  {
30210 '     inline
30220 '          $2a,#.a_l,       /* ld   hl,(.a_l)        */
30230 '          $ed,$5b,#.b_l,   /* ld   de,(.b_l)        */
30240 '          $19,             /* add  hl,de            */
30250 '          $dd,$75,$00,     /* ld   (ix),l           */
30260 '          $dd,$74,$01;     /* ld   (ix+1),h         */
30270 '  }
30280 'sub_w(a_l,a_h,b_l,b_h;ix,#);
30290 '  {
30300 '     inline
30310 '          $2a,#.a_l,       /* ld   hl,(.a_l)        */
30320 '          $ed,$5b,#.b_l,   /* ld   de,(.b_l)        */
30330 '          $b7,             /* or   a                */
30340 '          $ed,$52,         /* sbc  hl,de            */
30350 '          $dd,$75,$00,     /* ld   (ix),l           */
30360 '          $dd,$74,$01;     /* ld   (ix+1),h         */
30370 '  }
30380 'data _mul:00,_mul_w:00,00;
30390 'mul(a,b;#,#);
30400 '  var mul_m at(_code+6);
30410 '  {
30420 '    inline
30430 '          $21,#.b,         /* ld   hl,.b            */
30440 '          $3a,#.a,         /* ld   a,(.a)           */
30450 '          $cd,#.mul_m,     /* call mul_m            */
30460 '          $af,             /* xor  a                */
30470 '          $94,             /* sub  h                */
30480 '          $7c,             /* ld   a,h              */
30490 '          $32,#._mul,      /* ld   (._mul),a        */
30500 '          $7d;             /* ld   a,l              */
30510 '  }
30520 'mulov(;#,#);
30530 '  {
30540 '    _mul;
30550 '  }
30560 'mul_w(a_l,a_h,b_l,b_h;ix,#);
30570 '  {
30580 '    inline
30590 '          $c5,             /* push bc               */
30600 '          $21,#0000,       /* ld   hl,0000          */
30610 '          $ed,$4b,#.a_l,   /* ld   bc,(.a_l)        */
30620 '          $ed,$5b,#.b_l,   /* ld   de,(.b_l)        */
30630 '          $3e,$10,         /* ld   a,10h            */
30640 '          $29,             /* add  hl,hl            :lp      */
30650 '          $cb,$11,         /* rl   c                */
30660 '          $cb,$10,         /* rl   b                */
```

```
30670 '        $30,$04,        /* jr   nc,sl         */
30680 '        $19,            /* add  hl,de         */
30690 '        $30,$01,        /* jr   nc,sl         */
30700 '        $03,            /* inc  bc            */
30710 '        $3d,            /* dec  a             :sl      */
30720 '        $20,$f2,        /* jr   nz,lp         */
30730 '        $ed,$43,#._mul_w,/* ld   (._mul_w),bc  */
30740 '        $dd,$75,$00,    /* ld   (ix),l        */
30750 '        $dd,$74,$01,    /* ld   (ix+1),h      */
30760 '        $90,            /* sub  b             */
30770 '        $3e,$00,        /* ld   a,0           */
30780 '        $99,            /* sbc  a,c           */
30790 '        $c1;            /* pop  bc            */
30800 '  }
30810 'mulov_w(;ix,#);
30820 '  {
30830 '    @[ix,1]:=_mul_w[1];
30840 '    @[ix]:=_mul_w;
30850 '  }
30860 'data _mod:00,_mod_w:00,00;
30870 'div(a,b;#,#);
30880 '  var rem_m at(_code+$12);
30890 '  {
30900 '    inline
30910 '        $3a,#.a,        /* ld   a,(.a)        */
30920 '        $21,#.b,        /* ld   hl,.b         */
30930 '        $cd,#.rem_m,    /* call .rem_m        */
30940 '        $32,#._mod,     /* ld   (._mod),a     */
30950 '        $b7,            /* or   a             */
30960 '        $7b;            /* ld   a,e           */
30970 '  }
30980 'mod(;#,#);
30990 '  {
31000 '    _mod;
31010 '  }
31020 'div_w(a_l,a_h,b_l,b_h;ix,#);
31030 '  {
31040 '    inline
31050 '        $2a,#.a_l,      /* ld   hl,(.a_l)     */
31060 '        $ed,$4b,#.b_l,  /* ld   bc,(.b_l)     */
31070 '        $11,#0000,      /* ld   de,0000       */
31080 '        $3e,$10,        /* ld   a,10h         */
31090 '        $29,            /* add  hl,hl         :lp      */
31100 '        $eb,            /* ex   de,hl         */
31110 '        $ed,$6a,        /* adc  hl,hl         */
31120 '        $ed,$42,        /* sbc  hl,bc         */
31130 '        $13,            /* inc  de            */
31140 '        $30,$02,        /* jr   nc,sl         */
31150 '        $09,            /* add  hl,bc         */
31160 '        $1b,            /* dec  de            */
31170 '        $eb,            /* ex   de,hl         :sl      */
31180 '        $3d,            /* dec  a             */
31190 '        $20,$f1,        /* jr   nz,lp         */
31200 '        $dd,$75,00,     /* ld   (ix),l        */
31210 '        $dd,$74,01,     /* ld   (ix+1),h      */
31220 '        $ed,$53,#._mod_w,/* ld   (._mod_w),de  */
31230 '        $7b,            /* ld   a,e           */
31240 '        $b2;            /* or   d             */
31250 '  }
31260 'mod_w(;ix,#);
31270 '  {
31280 '    @[ix,1]:=_mod_w[1];
31290 '    @[ix]:=_mod_w;
31300 '  }
31310 'array(x,y,z;#ix,#);
31320 '  var  adr_l,adr_h;
31330 '  {
31340 '    if @[ix,4]-1
```

```
31350 '        then{
31360 '          adr_l:=mul(z,@[ix,3]);adr_h:=mulov();
31370 '          inline
31380 '           $2a,#.adr_l,     /* ld   hl,(.adr_l)     */
31390 '           $3a,#.y,         /* ld   a,(.y)          */
31400 '           $5f,             /* ld   e,a             */
31410 '           $16,$00,         /* ld   d,00            */
31420 '           $19,             /* add  hl,de           */
31430 '           $22,#.adr_l;     /* ld   hl,(.adr_l)     */
31440 '          mul_w(adr_l,adr_h,@[ix,2],0;.adr_l);
31450 '        }
31460 '        else{adr_l:=mul(y,@[ix,2]);adr_h:=mulov();}
31470 '      inline
31480 '           $2a,#.adr_l,     /* ld   hl,(.adr_l)     */
31490 '           $dd,$5e,00,      /* ld   e,(ix)          */
31500 '           $dd,$56,01,      /* ld   d,(ix+1)        */
31510 '           $19,             /* add  hl,de           */
31520 '           $3a,#.x,         /* ld   a,(.x)          */
31530 '           $5f,             /* ld   e,a             */
31540 '           $16,$00,         /* ld   d,00            */
31550 '           $19,             /* add  hl,de           */
31560 '           $e5,             /* push hl              */
31570 '           $dd,$e1;         /* pop  ix              */
31580 '  }
31590 'load_array(x,y,z;ix,#);
31600 '  {
31610 '     array(x,y,z;ix);
31620 '     @[ix];
31630 '  }
31640 'store_array(a,x,y,z;ix,#);
31650 '  {
31660 '     array(x,y,z;ix);
31670 '     @[ix]:=a;
31680 '  }
```

〔4〕ファイル入出力ライブラリ

これはディスク入出力のためのライブラリで，次の六つの関数を持っています。

（i） **open** (〈filenum〉，〈mode〉；〈ファイル名を示す文字列の先頭アドレス〉)

…BASICのOPEN文と同じ動作をします。〈filenum〉は0～2，〈mode〉は0：input，1：output，2：appendを示しています。〈ファイル名を示す文字列〉は二重引用符（″）で始まり，NULコードまたは二重引用符で終わるようにします。この open 関数ではランダム・アクセスはサポートしていません。

（ii） **close** （〈filenum〉)

…BASICのCLOSE文と同じでファイルをクローズするものです。

（iii） **output** （〈filenum〉，〈chr〉)

…〈filenum〉で指定したファイルへ〈chr〉を1文字出力します。

（iv） **input** （〈filenum〉)

…〈filenum〉で指定したファイルから1文字入力します。

（v） **eof** （〈filenum〉)

…〈filenum〉で指定したファイルが終わりに達したかどうか調べます。終わりなら ff Hを，そうでなければ0を返します。

（vi） **varptr** （〈filenum〉，〈fcbnum〉)

…〈filenum〉で指定したファイルのファイルコントロール・ブロックの〈fcbnum〉バイト目の値を返します。

●リスト3-4　　ファイル入出力ライブラリ

```
60000 '/*---------------------------------------
60010 ' *       written by H.Ohkuma
60020 ' *
60030 ' *                 09/05/84
60040 ' *---------------------------------------
60050 ' */
60060 '/*  open file
60070 ' *
60080 ' *     filenum[0..2]
60090 ' *     inout,in=0 out=1
60100 ' *     FileName
60110 ' */
60120 'open(filenum,inout;ix  /*FileName*/,#);
60130 '{
60140 '  if 2<inout then stop;
60150 '            else inout:=?(inout<2;inc(inout),8);
60160 '  inline
60170 '         $c5,            /* push   bc */
60180 '         $dd,$e5,        /* push   ix */
60190 '         $e1,            /* pop    hl */
60200 '         $cd,#$468c,     /* call   468ch */
60210 '         $3a,#.inout,    /* ld     a,(inout)*/
60220 '         $5f,            /* ld     e,a */
60230 '         $3a,#.filenum,  /* ld     a,(filenum)*/
60240 '         $cd,#$47f6,     /* call   47f6h */
60250 '         $21,#$0000,     /* ld     hl,0  */
60260 '         $22,#$ec88,     /* ld     (0ec88h),hl */
60270 '         $c1;            /* pop    bc  */
60280 '}
60290 '/*   close file
60300 ' *
60310 ' *      filenum[0..2]
60320 ' */
60330 'close(filenum;#,#);
60340 '{
60350 '  inline
60360 '         $c5,            /* push bc  */
60370 '         $3a,#.filenum,  /* ld   a,(filenum) */
60380 '         $cd,#$481d,     /* call 481dh */
60390 '         $21,#$0000,     /* ld   hl,0  */
60400 '         $22,#$ec88,     /* ld   (0ec88h),hl */
60410 '         $c1;            /* pop  bc  */
60420 '}
60430 '/*  output sequential data
60440 ' *
60450 ' *      filenum[0..2]
60460 ' *      a: output chracter
60470 ' */
60480 'output(filenum,a;#,#);
60490 '{
60500 '  inline
60510 '         $c5,            /* push bc     */
60520 '         $3a,#.filenum,  /* ld   a,(filenum) */
60530 '         $cd,#$4735,     /* call 4735h */
60540 '         $3a,#.a,        /* ld   a,(a) */
60550 '         $cd,#$4b54,     /* call 4b54h */
60560 '         $21,#$0000,     /* ld   hl,0  */
60570 '         $22,#$ec88,     /* ld   (0ec88h),hl */
60580 '         $c1;            /* pop  bc     */
60590 '}
60600 '/*  input sequential data
60610 ' *
60620 ' *      filenum[0..2]
60630 ' */
60640 'input(filenum;#,#);
60650 '{
60660 '  inline
```

```
60670 '           $c5,            /* push bc     */
60680 '           $3a,#.filenum,  /* ld   a,(filenum) */
60690 '           $cd,#$4735,     /* call 4735h */
60700 '           $cd,#$4b7b,     /* call 4b7bh */
60710 '           $21,#$0000,     /* ld   hl,0  */
60720 '           $22,#$ec88,     /* ld   (0ec88h),hl */
60730 '           $c1;            /* pop  bc     */
60740 '}
60750 '/*  end of file
60760 ' *
60770 ' *      filenum[0..2]
60780 ' */
60790 'eof(filenum;#,#);
60800 '{
60810 '  inline
60820 '          $c5,            /* push bc     */
60830 '          $3a,#.filenum,  /* ld   a,(filenum)      */
60840 '          $6f,            /* ld   l,a     */
60850 '          $26,$00,        /* ld   h,0     */
60860 '          $cd,#$21fd,     /* call 21fdh   */
60870 '          $3a,#$ec7d,     /* ld   a,(ec7dh) */
60880 '          $b7,            /* or   a       */
60890 '          $28,$05,        /* jr   z,nodisk*/
60900 '          $cd,#$a685,     /* call a685h   */
60910 '          $18,$03,        /* jr   skip    */
60920 '          $cd,#$4c51,     /*nodisk:call 4c51h */
60930 '          $3a,#$ec41,     /*skip:ld a,(ec41h) */
60940 '          $c1;            /*pop   bc      */
60950 '}
60960 '/*  varptr(#n)
60970 ' *
60980 ' *      filenum[0..2]
60990 ' *
61000 ' */
61010 'varptr(filenum,fcbnum;#,#);
61020 '{
61030 '  inline
61040 '           $c5,            /* push bc     */
61050 '           $3a,#.filenum,  /* ld   a,(filenum) */
61060 '           $cd,#$46f8,     /* call 46f8h  */
61070 '           $3a,#.fcbnum,   /* ld   a,(fcbnum) */
61080 '           $5f,            /* ld   e,a    */
61090 '           $16,$00,        /* ld   d,0    */
61100 '           $19,            /* add  hl,de  */
61110 '           $7e,            /* ld   a,(hl) */
61120 '           $c1;            /* pop  bc     */
61130 '}
```

〔5〕物理ディスク入出力ライブラリ

ディスクからの直接読み出しや書き込みをする関数のライブラリです。

（i）**dski**（<drive>, <track>, <surface>, <top sector>, <# of sector>；<buffer address>）

（ii）**dsko**（同上）

…dskiは読み出し，dskoは書き込みをする関数です。<drive>はドライブ番号，<track>はトラック番号，<surface>はヘッド番号で，<top sector>は読み出しあるいは書き込む初めのセクタ番号，<# of sector>は読み出しや書き込むセクタ数，<buffer address>は読み出したデータを記憶するバッファ，または書き込むデータを記憶している領域のアドレスです。

●リスト3-5　物理ディスク入出力ライブラリ

```
64000 '/*------------------------------------
64010 ' *     written by H.Ohkuma
64020 ' *
64030 ' *              09/05/84
64040 ' *------------------------------------
64050 ' */
64060 '/*
64070 ' *      dski : disk input
64080 ' *
64090 ' *        dr : drive number
64100 ' *        tr : track number
64110 ' *       sur : surface 0!1
64120 ' *       sec : sector number
64130 ' *   NoOfSec : number of sector
64140 ' *        ix : top address of read buffer
64150 ' *
64160 ' */
64170 'dski(dr,tr,sur,sec,NoOfSec;ix,#);
64180 'var DriveNo at ($ec85),
64190 '    DriveType at ($ef5d),
64200 '    ErrorCount at ($ecb4);
64210 '{
64220 '  inline
64230 '         $c5,              /* push bc      */
64240 '         $af,              /* xor  a       */
64250 '         $32,#.ErrorCount,/* ld   (EC),a */
64260 '         $3a,#.dr,         /* ld   a,(dr)  */
64270 '         $3d,              /* dec  a       */
64280 '         $32,#.DriveNo,    /* ld   (DN),a  */
64290 '         $cd,#$3dcb,       /* call 3dcbh   */
```

```
64300 '          $32,#.DriveType,/* ld   (DT),a  */
64310 '          $2a,#.tr,       /* ld   hl,(tr) */
64320 '          $7d,            /* ld   a,l     */
64330 '          $87,            /* add  a,a     */
64340 '          $84,            /* add  a,h     */
64350 '          $47,            /* ld   b,a     */
64360 '          $2a,#.sec,      /* ld   hl,(sec)*/
64370 '          $4d,            /* ld   c,l     */
64380 '          $7c,            /* ld   a,h     */
64390 '          $dd,$e5,        /* push ix      */
64400 '          $d1,            /* pop  de      */
64410 '          $b7,            /* or   a       */
64420 '          $cd,#$369d,     /* call 369dh   */
64430 '          $c1;            /* pop  bc      */
64440 '
64450 '  if carry() then stop;
64460 '}
64470 '/*
64480 ' *      dsko : disk output
64490 ' *
64500 ' *        dr : drive number
64510 ' *        tr : track number
64520 ' *       sur : surface 0|1
64530 ' *       sec : sector number
64540 ' *   NoOfSec : number of sector
64550 ' *        ix : top address of write buffer
64560 ' *
64570 ' */
64580 'dsko(dr,tr,sur,sec,NoOfSec;ix,#);
64590 'var DriveNo at ($ec85),
64600 '    DriveType at ($ef5d),
64610 '    ErrorCount at ($ecb4);
64620 '{
64630 '  inline
64640 '          $c5,            /* push bc      */
64650 '          $af,            /* xor  a       */
64660 '          $32,#.ErrorCount,/* ld   (EC),a */
64670 '          $3a,#.dr,       /* ld   a,(dr)  */
64680 '          $3d,            /* dec  a       */
64690 '          $32,#.DriveNo,  /* ld   (DN),a  */
64700 '          $cd,#$3dcb,     /* call 3dcbh   */
64710 '          $32,#.DriveType,/* ld   (DT),a  */
64720 '          $2a,#.tr,       /* ld   hl,(tr) */
64730 '          $7d,            /* ld   a,l     */
64740 '          $87,            /* add  a,a     */
64750 '          $84,            /* add  a,h     */
64760 '          $47,            /* ld   b,a     */
64770 '          $2a,#.sec,      /* ld   hl,(sec)*/
64780 '          $4d,            /* ld   c,l     */
64790 '          $7c,            /* ld   a,h     */
64800 '          $dd,$e5,        /* push ix      */
64810 '          $d1,            /* pop  de      */
64820 '          $37,            /* scf          */
64830 '          $cd,#$369d,     /* call 369dh   */
64840 '          $c1;            /* pop  bc      */
64850 '
64860 '  if carry() then stop;
64870 '}
```

3-5 サンプル・プログラム

ＰＣ-8801バージョンのStellarのサンプル・プログラムとして，ファイル・ダンプ，ファイル・タイプ，Stellarプログラムの圧縮，そしてゲームの四つのプログラムを示します。

〔1〕ファイル・ダンプ（リスト3-6）

ファイルの内容を16進数と文字で表示します。このプログラムは5インチ2Dのディスクでしか使えません。

●リスト3-6　ファイル・ダンプ

```
1000 '/*
1010 ' *       File Dump
1020 ' *
1030 ' *
1040 ' *     this program is for   PC-8801/mkII
1050 ' *                           & 5 inch 2D drive
1060 ' *
1070 ' *
1080 ' *            written by H.Ohkuma
1090 ' *
1100 ' *                      09/05/84
1110 ' */
1120 'prog dump();
1130 '
1140 'data firstmsg:"input file name = ",0,
1150 '      drive:"Drive:",0,
1160 '      track:"Track:",0,
1170 '      surface:"Surface:",0,
1180 '      sector:"Sector:",0;
1190 '
1200 'cons in:=0,
1210 '      out:=1,
1220 '      append:=2;
1230 '
1240 'var str[20],
1250 '     address[2],
1260 '     chrbuf[16],
1270 '     X at ($ef87),Y at ($ef86),
1280 '     i,dr,cl,sec;
1290 '{
1300 '  str:='"';
1310 '  print_str(;.firstmsg);
1320 '  input_str(20;.str+1); if carry() then stop;
1330 '  open(1,in;.str);
1340 '  address:=address[1]:=0;
1350 '  while not eof(1) {
1360 '    if address=0 then  {
```

```
1370 '
1380 '          chrbuf:=input(1); if eof(1) then { close(1); stop; }
1390 '
1400 '          dr:=varptr(1,4)+1;cl:=varptr(1,2);
1410 '          sec:=varptr(1,3);crlf(1);
1420 '          print_str(;.drive);print_chr(dr+$30);space(1);
1430 '          print_str(;.track);print_hex(cl/4);space(1);
1440 '          print_str(;.surface);print_hex((cl%4)/2);space(1);
1450 '          print_str(;.sector);print_hex(sec);crlf(1);
1460 '
1470 '          inp_data(1);
1480 '        } else { inp_data(0); }
1490 '
1500 '     print_hex(address[1]);print_hex(address);
1510 '     space(1);
1520 '     for i:=0 to 15 {
1530 '       print_hex(chrbuf[i]);
1540 '       space(1);
1550 '       inline $cd,#$5a86;
1560 '     }
1570 '     space(3);
1580 '     for i:=0 to 15 {
1590 '       poke_vram(X+i,Y,chrbuf[i]);
1600 '     }
1610 '     crlf(1);
1620 '     address:=address+16;address[1]:=address[1] plus 0;
1630 '   }
1640 '   close(1);
1650 '}
1660 'inp_data(a;#,#);
1670 '{
1680 '   for i:=a to 15 {
1690 '     chrbuf[i]:=input(1);
1700 '   }
1710 '}
1720 'poke_vram(x,y,c;#,#);
1730 '{
1740 '    inline
1750 '           $3a,#.x,        /* ld   a,(x)    */
1760 '           $67,            /* ld   h,a      */
1770 '           $3a,#.y,        /* ld   a,(y)    */
1780 '           $6f,            /* ld   l,a      */
1790 '           $cd,#$429d,     /* call 429dh    */
1800 '           $3a,#.c,        /* ld   a,(c)    */
1810 '           $77;            /* ld   (hl),a   */
1820 '}
1830 'space(num;#,#);
1840 'var i;
1850 '{
1860 '   if num=0 then return;
1870 '   for i:=1 to num {
1880 '     print_chr(' ');
1890 '   }
1900 '}
1910 '/*
1920 ' *I/O console
1930 ' */
1940 ' var  PRTFLG at($e64c);
1950 ' data _lpt:00;
1960 ' print_str(len;ix,#);
1970 '    {
1980 '      PRTFLG:=_lpt;
1990 '      inline
2000 '            $dd,$e5,       /* push ix       */
2010 '            $e1,           /* pop  hl       */
2020 '            $7e,           /*l1:ld   a,(hl) */
2030 '            $b7,           /* or   a        */
2040 '            $28,$04,       /* jp   z,$+6    */
```

```
2050 '          $df.          /* rst  18h      */
2060 '          $23.          /* inc  hl       */
2070 '          $18.$f8:      /* jr   11       */
2080 '    PRTFLG:=0:
2090 '  }
2100 ' print_hex(num:#.#):
2110 '  {
2120 '    PRTFLG:=_lpt:
2130 '    inline
2140 '          $3a.#.num,    /* ld   a,(.num)*/
2150 '          $e6,$f0.      /* and  f0h      */
2160 '          $07,          /* rlca          */
2170 '          $07,          /* rlca          */
2180 '          $07,          /* rlca          */
2190 '          $07,          /* rlca          */
2200 '          $cd,#._lbl_,  /* call ._lbl_   */
2210 '          $3a,#.num,    /* ld   a,(.num)*/
2220 '          $e6,$0f,      /* and  0fh      */
2230 '          $cd,#._lbl_;  /* call .lbl     */
2240 '    PRTFLG:=0;
2250 '    return;
2260 ' _lbl_:inline
2270 '          $c6,$30,      /* add  30h      */
2280 '          $fe,':',      /* cp   ':'      */
2290 '          $38,$02,      /* jr   c,02     */
2300 '          $c6,$07,      /* add  07h      */
2310 '          $df;          /* rst  18h      */
2320 '  }
2330 ' print_chr(chr:#,#);
2340 '  {
2350 '    PRTFLG:=_lpt;
2360 '    inline
2370 '          $3a,#.chr,    /* ld   a,(.chr)*/
2380 '          $df;          /* rst  18h      */
2390 '    PRTFLG:=0;
2400 '  }
2410 ' input_str(len;ix,#);
2420 '  {
2430 '    inline
2440 '          $c5,          /* push bc       */
2450 '          $cd,#$5fc8,   /* call 5fc8h    */
2460 '          $da,#._lbl_,  /* jp   c,._lbl_*/
2470 '          $3a,#.len,    /* ld   a,(.len)*/
2480 '          $47,          /* ld   b,a      */
2490 '          $0e,$00,      /* ld   c,00     */
2500 '          $05,          /* dec  b        */
2510 '          $dd,$e5,      /* push ix       */
2520 '          $d1,          /* pop  de       */
2530 '          $eb,          /* ex   de,hl    */
2540 '          $13,          /*lp:inc  de       */
2550 '          $1a,          /* ld   a,(de)  */
2560 '          $77,          /* ld   (hl),a  */
2570 '          $b7,          /* or   a        */
2580 '          $79,          /* ld   a,c      */
2590 '          $ca,#._lbl_,  /* jp   z,._lbl_*/
2600 '          $23,          /* inc  hl       */
2610 '          $0c,          /* inc  c        */
2620 '          $10,$f4,      /* djnz .lp      */
2630 '          $36,00,       /* ld   (hl),0  */
2640 '          $79;          /* ld   a,c      */
2650 ' _lbl_:inline
2660 '          $c1;          /* pop  bc       */
2670 '  }
2680 ' crlf(num;#,#);
2690 '  var i at(_work+10);
2700 '  cons CR:=13,LF:=10;
2710 '  {
2720 '    PRTFLG:=_lpt;
2730 '    for i:=1 to num {print_chr(CR);print_chr(LF);}
```

```
2740 '      PRTFLG:=0;
2750 '    }
2760 ' lpt_sw(sw;#,#);
2770 '   var LPTFLG at(._lpt);
2780 '   {
2790 '     if not(sw)
2800 '       then LPTFLG:=sw;
2810 '       else LPTFLG;
2820 '   }
2830 '/*  open file
2840 ' *
2850 ' *      filenum[0..2]
2860 ' *      inout.in=0 out=1
2870 ' *      FileName
2880 ' */
2890 'open(filenum,inout:ix  /*FileName*/,#):
2900 '{
2910 '  if 2<inout then stop:
2920 '           else inout:=?(inout<2;inc(inout),8);
2930 '  inline
2940 '         $c5,            /* push   bc */
2950 '         $dd,$e5,        /* push   ix */
2960 '         $e1,            /* pop    hl */
2970 '         $cd,#$468c,     /* call   468ch */
2980 '         $3a,#.inout,    /* ld     a,(inout)*/
2990 '         $5f,            /* ld     e,a */
3000 '         $3a,#.filenum,  /* ld     a,(filenum)*/
3010 '         $cd,#$47f6,     /* call   47f6h */
3020 '         $21,#$0000,     /* ld     hl,0  */
3030 '         $22,#$ec88,     /* ld     (0ec88h),hl */
3040 '         $c1;            /* pop    bc  */
3050 '}
3060 '/*  close file
3070 ' *
3080 ' *      filenum[0..2]
3090 ' */
3100 'close(filenum;#,#);
3110 '{
3120 '  inline
3130 '         $c5,            /* push bc  */
3140 '         $3a,#.filenum,  /* ld   a,(filenum) */
3150 '         $cd,#$481d,     /* call 481dh */
3160 '         $21,#$0000,     /* ld   hl,0  */
3170 '         $22,#$ec88,     /* ld   (0ec88h),hl */
3180 '         $c1;            /* pop  bc  */
3190 '}
3200 '/*  output sequential data
3210 ' *
3220 ' *      filenum[0..2]
3230 ' *      a: output chracter
3240 ' */
3250 'output(filenum,a;#,#);
3260 '{
3270 '  inline
3280 '         $c5,            /* push bc    */
3290 '         $3a,#.filenum,  /* ld   a,(filenum) */
3300 '         $cd,#$4735,     /* call 4735h */
3310 '         $3a,#.a,        /* ld   a,(a) */
3320 '         $cd,#$4b54,     /* call 4b54h */
3330 '         $21,#$0000,     /* ld   hl,0  */
3340 '         $22,#$ec88,     /* ld   (0ec88h),hl */
3350 '         $c1;            /* pop  bc    */
3360 '}
3370 '/*  input sequential data
3380 ' *
3390 ' *      filenum[0..2]
3400 ' */
3410 'input(filenum;#,#);
3420 '{
```

```
3430 '  inline
3440 '          $c5,            /* push bc     */
3450 '          $3a,#.filenum,  /* ld   a,(filenum) */
3460 '          $cd,#$4735,     /* call 4735h */
3470 '          $cd,#$4b7b,     /* call 4b7bh */
3480 '          $21,#$0000,     /* ld   hl,0  */
3490 '          $22,#$ec88,     /* ld   (0ec88h),hl */
3500 '          $c1;            /* pop  bc     */
3510 '}
3520 '/*  end of file
3530 ' *
3540 ' *      filenum[0..2]
3550 ' */
3560 'eof(filenum:#,#);
3570 '{
3580 '  inline
3590 '         $c5,            /* push bc      */
3600 '         $3a,#.filenum,  /* ld   a,(filenum)      */
3610 '         $6f,            /* ld   l,a      */
3620 '         $26,$00,        /* ld   h,0      */
3630 '         $cd,#$21fd,     /* call 21fdh    */
3640 '         $3a,#$ec7d,     /* ld   a,(ec7dh) */
3650 '         $b7,            /* or   a        */
3660 '         $28,$05,        /* jr   z,nodisk*/
3670 '         $cd,#$a685,     /* call a685h    */
3680 '         $18,$03,        /* jr   skip     */
3690 '         $cd,#$4c51,     /*nodisk:call 4c51h */
3700 '         $3a,#$ec41,     /*skip:ld a,(ec41h) */
3710 '         $c1;            /*pop   bc       */
3720 '}
3730 '/*  varptr(#n)
3740 ' *
3750 ' *      filenum[0..2]
3760 ' *
3770 ' */
3780 'varptr(filenum,fcbnum:#,#);
3790 '{
3800 '  inline
3810 '          $c5,            /* push bc     */
3820 '          $3a,#.filenum.  /* ld   a,(filenum) */
3830 '          $cd,#$46f8,     /* call 46f8h  */
3840 '          $3a,#.fcbnum,   /* ld   a,(fcbnum) */
3850 '          $5f,            /* ld   e,a     */
3860 '          $16,$00,        /* ld   d,0     */
3870 '          $19.            /* add  hl,de  */
3880 '          $7e,            /* ld   a,(hl) */
3890 '          $c1;            /* pop  bc      */
3900 '}
```

〔2〕ファイル・タイプ（リスト3−7）

ＡＳＣＩＩ形式のファイルの内容を表示します。ただし，表示できるファイルはＡＳＣＩＩ形式のファイルだけです。

●リスト3-7　　ファイル・タイプ

```
1000 '/*
1010 ' *       File Type
1020 ' *
1030 ' *
1040 ' *     this program is for   PC-8801/mkII
1050 ' *                           & 5 inch 2D drive
1060 ' *
1070 ' *
1080 ' *                written by H.Ohkuma
1090 ' *
1100 ' *                           09/05/84
1110 ' */
1120 'prog type();
1130 '
1140 'data firstmsg:"input file name = ",0,
1150 '     errmsg:"it's not ASCII type !!",7,0;
1160 '
1170 'cons in:=0,
1180 '     out:=1,
1190 '     append:=2;
1200 '
1210 'var str[20],
1220 '    c;
1230 '{
1240 '  str:='"';
1250 '  print_str(;.firstmsg);
1260 '  input_str(20;.str+1); if carry() then stop;
1270 '  open(1,in;.str);
1280 '
1290 '  if (varptr(1,7) and $81) then { print_str(;.errmsg); close(1); stop; }
1300 '  while not eof(1) {
1310 '     c:=input(1);
1320 '     print_chr(c);
1330 '     inline $cd,#$5a86;
1340 '  }
1350 '  close(1);
1360 '}
1370 '/*
1380 ' *I/O console
1390 ' */
1400 ' var  PRTFLG at($e64c);
1410 ' data _lpt:00;
1420 ' print_str(len;ix,#);
1430 '   {
1440 '     PRTFLG:=_lpt;
1450 '     inline
1460 '          $dd,$e5,       /* push ix      */
1470 '          $e1,           /* pop  hl      */
1480 '          $7e,           /*l1:ld   a,(hl) */
1490 '          $b7,           /* or   a       */
1500 '          $28,$04,       /* jp   z,$+6   */
1510 '          $df,           /* rst  18h     */
1520 '          $23,           /* inc  hl      */
```

```
1530 '          $18,$f8;      /* jr   l1      */
1540 '     PRTFLG:=0;
1550 '   }
1560 ' print_chr(chr;#,#);
1570 '   {
1580 '     PRTFLG:=_lpt;
1590 '     inline
1600 '          $3a,#.chr,    /* ld   a,(.chr)*/
1610 '          $df;          /* rst  18h     */
1620 '     PRTFLG:=0;
1630 '   }
1640 ' input_str(len;ix,#);
1650 '   {
1660 '     inline
1670 '          $c5,          /* push bc      */
1680 '          $cd,#$5fc8,   /* call 5fc8h   */
1690 '          $da,#._lbl_,  /* jp   c,._lbl_*/
1700 '          $3a,#.len,    /* ld   a,(.len)*/
1710 '          $47,          /* ld   b,a     */
1720 '          $0e,$00,      /* ld   c,00    */
1730 '          $05,          /* dec  b       */
1740 '          $dd,$e5,      /* push ix      */
1750 '          $d1,          /* pop  de      */
1760 '          $eb,          /* ex   de,hl   */
1770 '          $13,          /*lp:inc  de      */
1780 '          $1a,          /* ld   a,(de)  */
1790 '          $77,          /* ld   (hl),a  */
1800 '          $b7,          /* or   a       */
1810 '          $79,          /* ld   a,c     */
1820 '          $ca,#._lbl_,  /* jp   z,._lbl_*/
1830 '          $23,          /* inc  hl      */
1840 '          $0c,          /* inc  c       */
1850 '          $10,$f4,      /* djnz .lp     */
1860 '          $36,00,       /* ld   (hl),0  */
1870 '          $79;          /* ld   a,c     */
1880 ' _lbl_:inline
1890 '          $c1;          /* pop  bc      */
1900 '   }
1910 ' lpt_sw(sw;#,#);
1920 '   var LPTFLG at(._lpt);
1930 '   {
1940 '     if not(sw)
1950 '       then LPTFLG:=sw;
1960 '       else LPTFLG;
1970 '   }
1980 '/*  open file
1990 ' *
2000 ' *      filenum[0..2]
2010 ' *      inout,in=0 out=1
2020 ' *      FileName
2030 ' */
2040 'open(filenum,inout;ix  /*FileName*/,#);
2050 '{
2060 '  if 2<inout then stop;
2070 '            else inout:=?(inout<2;inc(inout),8);
2080 '  inline
2090 '          $c5,            /* push   bc */
2100 '          $dd,$e5,        /* push   ix */
2110 '          $e1,            /* pop    hl */
2120 '          $cd,#$468c,     /* call   468ch */
2130 '          $3a,#.inout,    /* ld     a,(inout)*/
2140 '          $5f,            /* ld     e,a */
2150 '          $3a,#.filenum,  /* ld     a,(filenum)*/
2160 '          $cd,#$47f6,     /* call   47f6h */
2170 '          $21,#$0000,     /* ld     hl,0  */
2180 '          $22,#$ec88,     /* ld     (0ec88h),hl */
2190 '          $c1;            /* pop    bc  */
2200 '}
2210 '/*  close file
```

```
2220 ' *
2230 ' *      filenum[0..2]
2240 ' */
2250 'close(filenum;#,#);
2260 '{
2270 '  inline
2280 '          $c5,              /* push bc  */
2290 '          $3a,#.filenum,    /* ld   a,(filenum) */
2300 '          $cd,#$481d,       /* call 481dh */
2310 '          $21,#$0000,       /* ld   hl,0  */
2320 '          $22,#$ec88,       /* ld   (0ec88h),hl */
2330 '          $c1;              /* pop  bc  */
2340 '}
2350 '/*  output sequential data
2360 ' *
2370 ' *      filenum[0..2]
2380 ' *      a: output chracter
2390 ' */
2400 'output(filenum,a;#,#);
2410 '{
2420 '  inline
2430 '          $c5,              /* push bc    */
2440 '          $3a,#.filenum,    /* ld   a,(filenum) */
2450 '          $cd,#$4735,       /* call 4735h */
2460 '          $3a,#.a,          /* ld   a,(a) */
2470 '          $cd,#$4b54,       /* call 4b54h */
2480 '          $21,#$0000,       /* ld   hl,0  */
2490 '          $22,#$ec88,       /* ld   (0ec88h),hl */
2500 '          $c1;              /* pop  bc    */
2510 '}
2520 '/*  input sequential data
2530 ' *
2540 ' *      filenum[0..2]
2550 ' */
2560 'input(filenum;#,#);
2570 '{
2580 '  inline
2590 '          $c5,              /* push bc    */
2600 '          $3a,#.filenum,    /* ld   a,(filenum) */
2610 '          $cd,#$4735,       /* call 4735h */
2620 '          $cd,#$4b7b,       /* call 4b7bh */
2630 '          $21,#$0000,       /* ld   hl,0  */
2640 '          $22,#$ec88,       /* ld   (0ec88h),hl */
2650 '          $c1;              /* pop  bc    */
2660 '}
2670 '/*  end of file
2680 ' *
2690 ' *      filenum[0..2]
2700 ' */
2710 'eof(filenum;#,#);
2720 '{
2730 '  inline
2740 '          $c5,              /* push bc     */
2750 '          $3a,#.filenum,    /* ld   a,(filenum)      */
2760 '          $6f,              /* ld   l,a      */
2770 '          $26,$00,          /* ld   h,0      */
2780 '          $cd,#$21fd,       /* call 21fdh   */
2790 '          $3a,#$ec7d,       /* ld   a,(ec7dh) */
2800 '          $b7,              /* or   a        */
2810 '          $28,$05,          /* jr   z,nodisk*/
2820 '          $cd,#$a685,       /* call a685h   */
2830 '          $18,$03,          /* jr   skip    */
2840 '          $cd,#$4c51,       /*nodisk:call 4c51h */
2850 '          $3a,#$ec41,       /*skip:ld a,(ec41h) */
2860 '          $c1;              /*pop   bc       */
2870 '}
2880 '/*  varptr(#n)
2890 ' *
2900 ' *      filenum[0..2]
```

```
2910 ' *
2920 ' */
2930 'varptr(filenum,fcbnum;#,#);
2940 '{
2950 '  inline
2960 '          $c5,            /* push bc    */
2970 '          $3a,#.filenum,  /* ld   a,(filenum) */
2980 '          $cd,#$46f8,     /* call 46f8h  */
2990 '          $3a,#.fcbnum,   /* ld   a,(fcbnum) */
3000 '          $5f,            /* ld   e,a    */
3010 '          $16,$00,        /* ld   d,0    */
3020 '          $19,            /* add  hl,de  */
3030 '          $7e,            /* ld   a,(hl) */
3040 '          $c1;            /* pop  bc     */
3050 '}
```

〔3〕Stellarプログラムの圧縮（リスト3−8）

Stellarのソース・プログラムのファイルを圧縮します。ライブラリを圧縮しておくと，使用メモリを小さくできフリーエリアが増します。ただし，ASCIIセーブされたソース・プログラムは圧縮できません。

●リスト3-8　　プログラムの圧縮

```
1000 '/*                                          */
1010 '/*          written by K.Ishizuka           */
1020 '/*                  09/05/84                */
1030 '/*                                          */
1040 'prog compressor();
1050 '  var LINK[2],LN[2],bf[256],
1060 '      c,len,i,err,
1070 '      filename1[20],filename2[20],
1080 '      SLASH,DQUOT,brace[2];
1090 '  data s1:"input source file name  : ",00,
1100 '       s2:"input destin file name  : ",00,
1110 '       s3:"input first line number : ",00;
1120 '  {
1130 '    print_str(;.s1);input_str(19;.filename1+1);if carry() then stop;
1140 '    print_str(;.s2);input_str(19;.filename2+1);if carry() then stop;
1150 '    print_str(;.s3);err:=input_num(2;.LN);
1160 '
1170 '    if err<>0 then if err=1 then stop;else error_trap(1);
1180 '    if LN=0 and LN[1]=0 then error_trap(2);
1190 '
1200 '    filename1:=filename2:='"';
1210 '    open(1,0;.filename1);open(2,1;.filename2);
1220 '    if(varptr(1,7) and $80)=0 then error_trap(3);
1230 '
1240 '    LINK:=LINK[1]:=DQUOT:=SLASH:=brace:=brace[1]:=0;
1250 '    erase_line_number(1);
1260 '    lp:
1270 '        len:=2;bf[len]:=LN;bf[inc(len)]:=LN[1];
1280 '        bf[inc(len)]:=$3a;bf[inc(len)]:=$8f;bf[inc(len)]:=$e9;
1290 '        if DQUOT then bf[inc(len)]:='"';
1300 '          {/*repeat*/
```

```
1310 '              c:=in(1);
1320 '              if DQUOT then bf[inc(len)]:=c;
1330 '                else{
1340 '                  if c='*' and SLASH then erase_comment();
1350 '                    elseif c=$27
1360 '                      then{
1370 '                        bf[inc(len)]:=c;
1380 '                        bf[inc(len)]:=in(1);bf[inc(len)]:=in(1);
1390 '                      }
1400 '                    elseif punc(c) and bf[len]=' ' then bf[len]:=c;
1410 '                    elseif not(punc(bf[len])&c=' ')then bf[inc(len)]:=c;
1420 '                 SLASH:=?(c='/';-1,0);
1430 '                 if c='{'then add_w(brace,brace[1],1,0;.brace);
1440 '                   elseif c='}' then add_w(brace,brace[1],-1,-1;.brace);
1450 '                }
1460 '              DQUOT:=?(c='"';not DQUOT,DQUOT);
1470 '            } until lend();
1480 '          if DQUOT then{bf[inc(len)]:='"';bf[inc(len)]:=',';}
1490 '          bf[inc(len)]:=00;
1500 '          add_w(LINK,LINK[1],len+1,0;.LINK);
1510 '          bf:=LINK;bf[1]:=LINK[1];
1520 '          for i:=0 to len{output(2,bf[i]);}
1530 '          add_w(LN,LN[1],10,0;.LN);if carry() then error_trap(2);
1540 '     goto lp;
1550 '   }
1560 'lend();
1570 '   {
1580 '     (len>=230&
1590 '        (punc(c)&~(SLASH|c=':'|('"'<=c&c<='$')|c=$27|c='<'|c='>')))|
1600 '     (len>=237&DQUOT)|
1610 '     ?(c='}';(brace[1]|brace)=0,0);
1620 '    }
1630 'in(fn);
1640 '   var a;
1650 '   {
1660 '     a:=input(fn);
1670 '     if a=00 then{a:=erase_line_number(fn);}
1680 '   a;}
1690 'erase_line_number(fn);
1700 '   var a[2],i;
1710 '   {
1720 '     a:=input(fn);a[1]:=input(fn);if a=0 and a[1]=0 then input_past_end();
1730 '     input(fn);input(fn);
1740 '     {;}until input(fn)=$e9;
1750 '     $20;
1760 '   }
1770 'punc(a);
1780 '   {
1790 '     if a<='/' then -1;
1800 '       elseif ':'<=a and a<='?' then -1;
1810 '       elseif '['<=a and a<='^' then -1;
1820 '       elseif '{'<=a and a<='~' then -1;
1830 '       elseif a=$e9 then -1;else 0;
1840 '   }
1850 'input_past_end();
1860 '   var i;
1870 '   {
1880 '     bf[inc(len)]:=0;
1890 '     add_w(LINK,LINK[1],len+1,0;.bf);
1900 '     bf[inc(len)]:=bf[inc(len)]:=0;
1910 '     for i:=0 to len{output(2,bf[i]);}
1920 '     for i:=0 to 2{output(2,-1);}
1930 '     close(1);close(2);
1940 '     setbasfil(;.filename2);
1950 '     stop;
1960 '   }
1970 'erase_comment();
1980 '   var a,b;
1990 '   { a:=b:=0;
```

```
2000 '    {{a:=b;b:=input(1);}until a='*';}until b='/';
2010 '    SLASH:=0;dec(len);
2020 '  }
2030 'setbasfil(;ix,#);
2040 '  {
2050 '    inline
2060 '          $dd,$e5,         /* push ix      */
2070 '          $e1,             /* pop  hl      */
2080 '          $cd,#$468c,      /* call 468ch   */
2090 '          $cd,#$4742,      /* call 4742h   */
2100 '          $cd,#$98ea,      /* call 98eah   */
2110 '          $11,#$0009,      /* ld   de,0009h*/
2120 '          $19,             /* add  hl,de   */
2130 '          $36,$80,         /* ld   (hl),80h*/
2140 '          $3a,#$ec8d,      /* ld   a,(ec8dh)*/
2150 '          $57,             /* ld   d,a     */
2160 '          $cd,#$a340;      /* call 0a340h  */
2170 '  }
2180 'error_trap(err);
2190 '  data e1:7,"Input error",13,10,0,
2200 '       e2:7,"Line number OV error",13,10,0,
2210 '       e3:7,"Bad sorce error",13,10,0;
2220 '  {
2230 '    if err=1 then print_str(;.e1);
2240 '    if err=2 then{print_str(;.e2);close(1);close(2);}
2250 '    if err=3 then{print_str(;.e3);close(1);close(2);}
2260 '    stop;
2270 '  }
2280 'var PRTFLG at($e64c);data _lpt:00;print_str(len;ix,#);{PRTFLG:=_lpt;inline
$dd,$e5,$e1,$7e,$b7,$28,$04,$df,$23,$18,$f8;PRTFLG:=0;}
2290 'input_str(len;ix,#);{inline$c5,$cd,#$5fc8,$38,$19,$3a,#.len,$47,$0e,$00,$0
5,$dd,$e5,$d1,$eb,$13,$1a,$77,$b7,$79,$28,$07,$23,$0c,$10,$f5,$36,$00,$79,$c1;}
2300 'input_num(type;ix,#);data _lbl_:$cd,#$5fc8,$3e,$01,$d8,$23,$cd,#$26bc,$f7,
$3e,$02,$f0,$2a,#$ec41,$3a,#.type,$dd,$75,$00,$3d,$28,$05,$dd,$74,$01,$af,$c9,$7
c,$b7,$c8,$3e,$02,$37,$c9;{inline$cd,#._lbl_;}
2310 '
2320 'add_w(a_l,a_h,b_l,b_h;ix,#);{inline$2a,#.a_l,$ed,$5b,#.b_l,$19,$dd,$75,$00
,$dd,$74,$01;}
2330 'open(filenum,inout;ix,#);{if 2<inout then stop;else inout:=?(inout<2;inc(i
nout),8);inline$c5,$dd,$e5,$e1,$cd,#$468c,$3a,#.inout,$5f,$3a,#.filenum,$cd,#$47
f6,$21,#$0000,$22,#$ec88,$c1;}
2340 'close(filenum;#,#);{inline$c5,$3a,#.filenum,$cd,#$481d,$21,#$0000,$22,#$ec
88,$c1;}
2350 'output(filenum,a;#,#);{inline$c5,$3a,#.filenum,$cd,#$4735,$3a,#.a,$cd,#$4b
54,$21,#$0000,$22,#$ec88,$c1;}
2360 'input(filenum;#,#);{inline$c5,$3a,#.filenum,$cd,#$4735,$cd,#$4b7b,$21,#$00
00,$22,#$ec88,$c1;}
2370 'varptr(filenum,fcbnum;#,#);{inline$c5,$3a,#.filenum,$cd,#$46f8,$3a,#.fcbnu
m,$5f,$16,$00,$19,$7e,$c1;}
2380 '
```

〔4〕サンプル・ゲーム（リスト3-9）

コンパイルされたプログラムの速度をみるためにリアルタイム・ゲームをつくってみました。遊び方は，テンキーの8，2，4，6で，上下左右に操り，テンキーとスペース・バーを同時に押してブロックを移動します。敵に捕まらないようにブロックで敵をやっつけてください。

なお，リストで空白のない部分は，〔3〕のプログラムで圧縮したライブラリをマージしたものです。

●リスト3-9　　サンプル・ゲーム

```
1000 '/*                                               */
1010 '/*          written by K.Ishizuka                */
1020 '/*                    09/05/84                   */
1030 '/*                                               */
1040 'prog sample_game();
1050 ' {main();}
1060 'console(s,n,F,C;#,#);var CNSDFG at($e6b8),LINEND at($e6b1);var SCRLL1 at($
e6b2),SCRLL2 at($e6b3);var CMODE at($e6b9);{SCRLL1:=?(s<=25;s+1,?(not s;25,SCRLL
1));SCRLL2:=?(n<=25;?(n;n+s,1),?(not n;25,SCRLL2));if not F then{if F then
1070 'CNSDFG:=$ff;else{inline$cd,#$4021;CNSDFG:=0;}}C:=?(not C;C,CMODE);inline$3
a,#.C,$cd,#$70d1;}
1080 'width(h,v;#,#);var LINCNT at($ef88),LINWDT at($ef89);{if not h then;else h
:=LINWDT;if not v then;else v:=LINCNT;if(h=40 or h=80)and(v=25 or v=20)then inli
ne$2a,#.h,$45,$4c,$cd,#$6f6b;}
1090 'screen(c,f;#,#);var HIRESL at($e6a6),PORT31 at($e6c2);var PORT40 at($e6c1)
;{if inc(c)then{if dec(c)then{if dec(c)then{if dec(c)then;else{HIRESL:=1;port[$3
1]:=PORT31:=PORT31 and$ee;}}else{HIRESL:=0;port[$31]:=PORT31:=(PORT31 or$01)
1100 'and$ef;}}else{HIRESL:=0;port[$31]:=PORT31:=PORT31 or$11;}}if not f then{if
 f and$01 then port[$40]:=PORT40:=PORT40 or$10;else port[$40]:=PORT40:=PORT40 an
d$ef;if f and$02 then port[$31]:=PORT31:=PORT31 and$f7;else port[$31]:=PORT31:=
1110 'PORT31 or$08;}}
1120 'cls(n;#,#);{if n and$01 then inline$cd,#$5f0e;if n and$02 then inline$f3,$
3a,#$e6c1,$f6,$10,$d3,$40,$3e,$5c,$4f,$ed,$79,$01,#$3e7f,$11,#$c001,$21,#$c000,$
36,$00,$ed,$b0,$3c,$fe,$5f,$20,$eb,$d3,$5f,$3a,#$e6c1,$d3,$40,$fb;}
1130 'locate(X,Y,F;#,#);var CSRY at($ef86),CSRX at(.CSRY+1);var LINCNT at(.CSRY+
2),LINWDT at(.CSRY+3);var CURFG at($e6a7);{CSRY:=?(LINCNT>Y;Y+1,LINCNT);CSRX:=?(
LINWDT>X;X+1,LINWDT);if not F then CURFG:=?(F;1,0);}
1140 '
1150 'put(X,Y;IX);
1160 '{
1170 ' inline
1180 ' $f3,
1190 ' $3a,#.Y,
1200 ' $5f,
1210 ' $87,
1220 ' $6f,
1230 ' $26,$00,
1240 ' $54,
1250 ' $29,
1260 ' $19,
1270 ' $29,
1280 ' $29,
1290 ' $29,
1300 ' $29,
1310 ' $29,
1320 ' $29,
```

```
1330 ' $3a,#.X,
1340 ' $5f,
1350 ' $19,
1360 ' $11,#$c000,
1370 ' $19,
1380 ' $af,
1390 ' $eb,
1400 ' $dd,$e5,
1410 ' $e1,
1420 ' $23,
1430 ' $23,
1440 ' $01,$5c,$03,
1450 ' $c5,
1460 ' $d5,
1470 ' $dd,$7e,$01,
1480 ' $87,
1490 ' $87,
1500 ' $47,
1510 ' $af,
1520 ' $c5,
1530 ' $d5,
1540 ' $ed,$79,
1550 ' $dd,$4e,$00,
1560 ' $47,
1570 ' $ed,$b0,
1580 ' $d3,$5f,
1590 ' $d1,
1600 ' $eb,
1610 ' $01,$50,$00,
1620 ' $09,
1630 ' $eb,
1640 ' $c1,
1650 ' $10,$ea,
1660 ' $d1,
1670 ' $c1,
1680 ' $0c,
1690 ' $10,$dc,
1700 ' $fb;
1710 ' }
1720 'data ch_block1:$2,$2,
1730 ' $00,$80,$00,$80,$00,$ff,$ff,/*B*/
1740 ' $00,$01,$00,$01,$00,$01,$ff,$ff,
1750 ' $aa,$aa,$55,$d5,$aa,$aa,$ff,$ff,/*R*/
1760 ' $aa,$ab,$55,$55,$aa,$ab,$ff,$ff,
1770 ' $00,$80,$00,$80,$00,$80,$ff,$ff,/*G*/
1780 ' $00,$01,$00,$01,$00,$01,$ff,$ff;
1790 'data ch_block2:$2,$2,
1800 ' $ff,$ff,$ff,$ff,$ff,$ff,$ff,$ff,/*B*/
1810 ' $ff,$ff,$ff,$ff,$ff,$ff,$ff,$ff,
1820 ' $aa,$aa,$55,$d5,$aa,$aa,$ff,$ff,/*R*/
1830 ' $aa,$ab,$55,$55,$aa,$ab,$ff,$ff,
1840 ' $00,$80,$00,$80,$00,$80,$ff,$ff,/*G*/
1850 ' $00,$01,$00,$01,$00,$01,$ff,$ff;
1860 'data ch_road:$2,$2,
1870 ' $00,$00,$00,$00,$00,$00,$00,$00,/*B*/
1880 ' $00,$00,$00,$00,$00,$00,$00,$00,
1890 ' $00,$00,$00,$00,$00,$00,$00,$00,/*R*/
1900 ' $00,$00,$00,$00,$00,$00,$00,$00,
1910 ' $00,$00,$00,$00,$00,$00,$00,$00,/*G*/
1920 ' $00,$00,$00,$00,$00,$00,$00,$00;
1930 'data ch_man1:$4,$4,
1940 ' $38,$03,$c0,$1c,$f8,$0f,$f0,$1f,/*B*/
1950 ' $d0,$18,$18,$06,$08,$18,$18,$28,
1960 ' $05,$1c,$18,$50,$02,$8f,$f2,$80,
1970 ' $01,$55,$55,$00,$00,$aa,$aa,$00,
1980 ' $00,$55,$54,$00,$00,$aa,$aa,$00,
1990 ' $00,$55,$54,$00,$00,$2e,$b8,$00,
2000 ' $01,$54,$14,$00,$02,$b0,$02,$80,
2010 ' $01,$00,$01,$00,$02,$80,$02,$80,
```

```
2020 ' $38,$03,$c0,$1c,$f8,$0f,$f0,$1f,/*R*/
2030 ' $f0,$1f,$f8,$0e,$1c,$19,$98,$38,
2040 ' $0f,$1f,$f8,$f0,$03,$cf,$f3,$c0,
2050 ' $01,$ff,$ff,$00,$20,$ff,$fe,$00,
2060 ' $40,$fa,$be,$00,$40,$f5,$5e,$00,
2070 ' $38,$6a,$ac,$00,$0e,$39,$58,$00,
2080 ' $01,$fd,$3e,$00,$03,$f0,$83,$80,
2090 ' $01,$9c,$43,$00,$07,$c3,$87,$c0,
2100 ' $00,$00,$00,$00,$18,$03,$c0,$18,/*G*/
2110 ' $10,$08,$10,$00,$00,$08,$10,$00,
2120 ' $00,$0c,$10,$00,$00,$03,$c0,$00,
2130 ' $00,$00,$00,$00,$00,$00,$00,$00,
2140 ' $00,$05,$40,$00,$00,$0a,$a0,$00,
2150 ' $00,$15,$50,$00,$00,$06,$a0,$00,
2160 ' $00,$00,$00,$00,$00,$00,$00,$00,
2170 ' $00,$00,$00,$00,$02,$80,$02,$80;
2180 'data ch_man2:$4,$4,
2190 ' $01,$c0,$03,$80,$07,$83,$c1,$e0,/*B*/
2200 ' $0c,$0f,$f0,$20,$0c,$18,$18,$50,
2210 ' $0a,$18,$18,$a0,$05,$1c,$19,$40,
2220 ' $02,$8c,$32,$c0,$01,$55,$55,$00,
2230 ' $00,$aa,$aa,$00,$00,$55,$54,$00,
2240 ' $00,$aa,$aa,$00,$00,$55,$54,$00,
2250 ' $00,$2e,$b8,$00,$01,$54,$15,$00,
2260 ' $02,$b0,$0a,$80,$00,$70,$0e,$00,
2270 ' $01,$c0,$03,$80,$07,$83,$c1,$e0,/*R*/
2280 ' $0c,$0f,$f0,$30,$0e,$1f,$f8,$70,
2290 ' $0f,$19,$98,$f0,$07,$9f,$f9,$e0,
2300 ' $03,$cc,$33,$c0,$01,$ff,$ff,$0c,
2310 ' $00,$ff,$fe,$02,$00,$fa,$be,$04,
2320 ' $00,$f5,$5e,$08,$00,$6a,$ac,$18,
2330 ' $00,$39,$58,$60,$01,$fd,$3f,$80,
2340 ' $03,$f0,$ef,$c0,$00,$f8,$1f,$00,
2350 ' $00,$00,$00,$00,$01,$80,$01,$80,/*G*/
2360 ' $00,$03,$c0,$00,$00,$08,$10,$00,
2370 ' $00,$08,$10,$00,$00,$0c,$10,$00,
2380 ' $00,$00,$00,$00,$00,$00,$00,$00,
2390 ' $00,$00,$00,$00,$00,$05,$40,$00,
2400 ' $00,$0a,$a0,$00,$00,$15,$50,$00,
2410 ' $00,$06,$a0,$00,$00,$00,$00,$00,
2420 ' $00,$00,$00,$00,$00,$70,$0e,$00;
2430 'data ch_enemy1:$4,$4,
2440 ' $00,$00,$00,$00,$00,$00,$00,$00,/*B*/
2450 ' $0c,$00,$00,$30,$30,$0e,$70,$0c,
2460 ' $0c,$71,$8e,$30,$03,$80,$01,$c0,
2470 ' $00,$00,$00,$00,$00,$00,$00,$00,
2480 ' $00,$00,$00,$00,$00,$00,$00,$00,
2490 ' $00,$08,$10,$00,$00,$04,$20,$00,
2500 ' $00,$1e,$78,$00,$00,$03,$c0,$00,
2510 ' $00,$00,$00,$00,$00,$00,$00,$00,
2520 ' $00,$00,$00,$00,$00,$00,$00,$00,/*R*/
2530 ' $0c,$00,$00,$30,$30,$0e,$70,$0c,
2540 ' $0c,$71,$8e,$30,$03,$80,$01,$c0,
2550 ' $00,$06,$60,$00,$00,$06,$60,$00,
2560 ' $00,$00,$00,$00,$00,$00,$00,$00,
2570 ' $00,$08,$10,$00,$00,$04,$20,$00,
2580 ' $00,$1e,$78,$00,$00,$03,$c0,$00,
2590 ' $00,$00,$00,$00,$00,$00,$00,$00,
2600 ' $00,$00,$00,$00,$00,$00,$00,$00,/*G*/
2610 ' $00,$00,$00,$00,$30,$0e,$70,$0c,
2620 ' $0c,$71,$8e,$30,$03,$8f,$f1,$c0,
2630 ' $00,$39,$9c,$00,$48,$39,$9c,$12,
2640 ' $90,$1f,$f0,$09,$b0,$04,$20,$0d,
2650 ' $70,$7f,$fe,$0e,$31,$c7,$e3,$8c,
2660 ' $0f,$1f,$f8,$78,$00,$e3,$c7,$00,
2670 ' $07,$00,$00,$e0,$38,$00,$00,$1c;
2680 'data ch_enemy2:$4,$4,
2690 ' $c0,$0c,$30,$03,$30,$32,$4c,$0c,/*B*/
2700 ' $0c,$c1,$83,$30,$03,$00,$00,$c0,
```

```
2710 ' $00,$00,$00,$00,$00,$00,$00,$00,
2720 ' $00,$00,$00,$00,$00,$00,$00,$00,
2730 ' $00,$08,$10,$00,$00,$04,$20,$00,
2740 ' $00,$1e,$78,$00,$00,$03,$c0,$00,
2750 ' $00,$00,$00,$00,$00,$00,$00,$00,
2760 ' $00,$00,$00,$00,$00,$00,$00,$00,
2770 ' $c0,$0c,$30,$03,$30,$32,$4c,$0c,/*R*/
2780 ' $0c,$c1,$83,$30,$03,$00,$00,$c0,
2790 ' $00,$06,$60,$00,$00,$06,$60,$00,
2800 ' $00,$00,$00,$00,$00,$00,$00,$00,
2810 ' $00,$08,$10,$00,$00,$04,$20,$00,
2820 ' $00,$1e,$78,$00,$00,$03,$c0,$00,
2830 ' $00,$00,$00,$00,$00,$00,$00,$00,
2840 ' $00,$00,$00,$00,$00,$00,$00,$00,
2850 ' $00,$0c,$30,$00,$30,$32,$4c,$0c,/*G*/
2860 ' $0c,$c1,$83,$30,$03,$0f,$f0,$c0,
2870 ' $00,$39,$9c,$00,$00,$39,$9c,$00,
2880 ' $0f,$1f,$f0,$f0,$31,$c4,$23,$8c,
2890 ' $70,$7f,$fe,$0e,$b0,$07,$e0,$0d,
2900 ' $90,$1f,$f8,$09,$48,$63,$c6,$12,
2910 ' $01,$80,$01,$80,$03,$00,$00,$c0,
2920 ' $03,$00,$00,$c0,$07,$00,$00,$e0;
2930 'var  map_p;
2940 'data map:
2950 ' 33,30,2,30,2,2,6,2,10,2,6,2,2,6,
2960 ' 2,6,2,6,2,6,2,6,2,6,2,6,2,6,6,2,
2970 ' 6,2,2,2,6,2,6,14,2,14,2,14,2,14,
2980 ' 2,2,8,2,6,2,8,2,2,8,2,10,2,8,2,
2990 ' 8,2,10,2,8,8,2,4,2,2,2,4,2,8,
3000 ' 14,2,14,2,14,2,14,2,2,10,2,2,2,
3010 ' 10,2,2,14,2,14,2,14,2,14,2,2,6,
3020 ' 2,2,2,2,2,2,2,6,2,2,10,2,6,2,10,
3030 ' 2,10,2,6,2,10,34,30,33,0;
3040 'data mess:" HI      0 SCORE      0 MONKEY 3",00;
3050 'var  init_p;
3060 'data init:
3070 ' 13,7,15,19,1,0,2,
3080 ' 21,4,23,1,1,0,10,
3090 ' 7,16,11,13,$ff,0,13,
3100 ' 27,16,15,1,$ff,0,7;
3110 'var hiscore[2],score[2],score_h at (.score+1),man,enemy;
3120 'var MX,MY,BoxX[4],BoxY[4],OL[4],OX[4],OY[4],DX[4],DY[4],DS[4];
3130 'var IV_w[704];data IV:#.IV_w,32,22,1;
3140 'var H,C,BackC,D,Z,K,V,W,End;
3150 'var i;
3160 'main();
3170 '{
3180 ' width(40,25);console(0,25,0,0);cls(3);screen(-1,0);
3190 ' make_stage();
3200 ' locate(1,22);print_str(;.mess);
3210 ' hiscore:=0;hiscore[1]:=0;
3220 '  {/*repeat*/
3230 '    man:=3;enemy:=1;
3240 '    score:=0;score[1]:=0;
3250 '     {/*repeat*/
3260 '       for i:=0 to 3 {OL[i]:=?(i<enemy;1,0);}
3270 '        {/*repeat*/
3280 '          init_val();init_stage();
3290 '           {/*repeat*/
3300 '             D:=move_enemy();
3310 '             if D=0
3320 '               then {
3330 '                 enemy:=?(enemy=4;1,enemy+1);
3340 '                 Q_put(MX*2,MY*2;.ch_road);box_clear();
3350 '               }
3360 '               else {
3370 '                 if C:=not(port[00] and port[01]) and $55
3380 '                   then {
3390 '                     if C and $01 then move_(0,-1);
```

```
3400 '                elseif C and $40 then move_(1,0);
3410 '                elseif C and $04 then move_(0,1);
3420 '                elseif C and $10 then move_(-1,0);
3430 '             }
3440 '             else loop i,255{loop #,40{;}}
3450 '            if K:=?(K=0;1,0)
3460 '              then put(MX*2,MY*2;.ch_man1);else put(MX*2,MY*2;.ch_man2);
3470 '            dead_check();
3480 '            if Z
3490 '             then {
3500 '              Q_put(MX*2,MY*2;.ch_road);
3510 '              sound(1);
3520 '              dec(man);
3530 '              locate(30,22,1);print_using(1,man,0);
3540 '              box_clear();enemy_clear();
3550 '              if man=0 then game_over();
3560 '             }
3570 '           }
3580 '        } until Z<>0 or D=0;
3590 '      } until man=0 or D=0;
3600 '    } until man=0;
3610 '  } until End=1;
3620 ' width(80,20);screen(-1,3);
3630 '}
3640 'make_stage();
3650 'var i,n;
3660 'var X,Y;
3670 '{
3680 ' map_p:=-1;X:=0;Y:=0;
3690 '  {/*repeat*/
3700 '   n:=map[inc(map_p)];
3710 '   if n
3720 '    then {
3730 '     for i:=1 to n
3740 '      {
3750 '       put(X,Y;.ch_block1);
3760 '       init_IV(X,Y,1);
3770 '       if (X:=(X+2)%64)=0 then Y:=Y+2;
3780 '      }
3790 '     n:=map[inc(map_p)];
3800 '     if n then
3810 '       for i:=1 to n
3820 '        {
3830 '         put(X,Y;.ch_road);
3840 '         init_IV(X,Y,0);
3850 '         if (X:=(X+2)%64)=0 then Y:=Y+2;
3860 '        }
3870 '    }
3880 '  } until n=0;
3890 '}
3900 'init_IV(X,Y,a);
3910 'var IV_adr[2];
3920 '{
3930 '  array(X/2,Y/2;.IV);
3940 '  if Y<=42 then memory[ix]:=a;
3950 '}
3960 'init_val();
3970 '{
3980 '  init_p:=-1;
3990 '  for i:=0 to 3
4000 '   {
4010 '    BoxX[i]:=init[inc(init_p)];
4020 '    BoxY[i]:=init[inc(init_p)];
4030 '    OX[i]:=init[inc(init_p)];OY[i]:=init[inc(init_p)];
4040 '    DX[i]:=init[inc(init_p)];DY[i]:=init[inc(init_p)];
4050 '    DS[i]:=init[inc(init_p)];
4060 '    if OL[i]=0 then OX[i]:=0;
4070 '   }
4080 '  MX:=15;MY:=10;H:=1;
```

```
4090 '}
4100 'init_stage();
4110 '{
4120 '  for i:=0 to 3 {put_(BoxX[i],BoxY[i],1:.ch_block2);}
4130 '  locate(4,22);print_using(5,hiscore,hiscore[1]);
4140 '  locate(16,22);print_using(5,score,score[1]);
4150 '  locate(30,22);print_using(1,man,0);
4160 '}
4170 'move_enemy();
4180 'var D,X,Y,V,W,Z;
4190 'var i;
4200 '{
4210 ' D:=0;
4220 ' H:=?(H=1;0,1);
4230 ' for i:=0 to 3
4240 '  {
4250 '   if OL[i] then
4260 '    {
4270 '     inc(D);
4280 '     Z:=move_a_enemy(OX[i],OY[i],DX[i],DY[i],i);
4290 '    }
4300 '  }
4310 ' D;
4320 '}
4330 'move_a_enemy(X,Y,V,W,i);
4340 'var Z,t,u,o;
4350 '{
4360 ' Z:=move_check(X,Y,V,W);
4370 ' if Z=0 then
4380 '  {
4390 '   t:=V;u:=W;o:=sgn();
4400 '   W:=t;V:=u;
4410 '   if o then
4420 '    {
4430 '     if V=0 then W:=o;else V:=o;
4440 '     Z:=move_check(X,Y,V,W);
4450 '     if Z=0 then
4460 '      {
4470 '       V:=-V;W:=-W;
4480 '       Z:=move_check(X,Y,V,W);
4490 '      }
4500 '    }
4510 '   if Z=0 then
4520 '    {
4530 '     V:=-t;W:=-u;
4540 '     Z:=move_check(X,Y,V,W);
4550 '     if Z=0 then
4560 '      {
4570 '       OL[i]:=0;
4580 '       if carry(score:=score+10) then inc(score_h);
4590 '       sound(0);
4600 '       locate(16,22);print_using(5,score,score[1]);
4610 '      }
4620 '    }
4630 '  }
4640 ' Q_put(OX[i]*2,OY[i]*2;.ch_road);
4650 ' if Z=1 then
4660 '  {
4670 '   OX[i]:=X+V;OY[i]:=Y+W;
4680 '   DX[i]:=V;DY[i]:=W;
4690 '   if H
4700 '     then put(OX[i]*2,OY[i]*2;.ch_enemy1);
4710 '     else put(OX[i]*2,OY[i]*2;.ch_enemy2);
4720 '  }
4730 ' Z;
4740 '}
4750 'move_(V,W);
4760 'var i;
4770 '{
```

```
4780 ' if not port[09] and $40
4790 '  then move_block(V,W);
4800 '  else {loop i,255{loop#,30{;}}Q_put(MX*2,MY*2;.ch_road);move_man(V,W);}
4810 '}
4820 'move_man(V,W);
4830 '{
4840 ' if move_check(MX,MY,V,W)=1
4850 '  then {
4860 '   MX:=MX+V;
4870 '   MY:=MY+W;
4880 '  }
4890 '}
4900 'move_block(V,W);
4910 'var i,X,Y;
4920 '{
4930 ' for i:=0 to 3 {move_a_block(BoxX[i],BoxY[i],V,W,i);}
4940 '}
4950 'move_a_block(X,Y,V,W,i);
4960 '{
4970 ' if move_check(X,Y,V,W)
4980 '  then {
4990 '   put_(X,Y,0;.ch_road);
5000 '   X:=X+V;Y:=Y+W;
5010 '   put_(X,Y,1;.ch_block2);
5020 '   BoxX[i]:=X;BoxY[i]:=Y;
5030 '  }
5040 '}
5050 'move_check(X,Y,V,W);
5060 'var p,q,r,s;
5070 '{
5080 ' p:=0;q:=0;
5090 ' if V+W<=$7f
5100 '  then {r:=V*2;s:=W*2;}
5110 '  else {r:=V;s:=W;}
5120 ' if V=0 then p:=1; else q:=1;
5130 ' r:=X+r;s:=Y+s;
5140 ' if load_array(r,s;.IV)+load_array(r+p,s+q;.IV) then 0;else 1;
5150 '}
5160 'dead_check();
5170 '{
5180 ' Z:=0;
5190 ' for i:=0 to 3 {if MX=OX[i] then if MY=OY[i] then Z:=1;}
5200 '}
5210 'game_over();
5220 'var key;
5230 '{
5240 ' inline $cd,#$35d9;
5250 ' {/*repeat*/ key:=input_chr();End:=?(key=$03;1,?(key=$20;2,0));
5260 ' }until End=1 or End=2;
5270 ' if compare_w(;.score,.hiscore) then move_w(;.hiscore,.score);
5280 '}
5290 'enemy_clear();
5300 'var i;
5310 '{
5320 ' for i:=0 to 3
5330 '  { if OL[i] then Q_put(OX[i]*2,OY[i]*2;.ch_road);}
5340 '}
5350 'box_clear();
5360 'var i;
5370 '{
5380 ' for i:=0 to 3 {put_(BoxX[i],BoxY[i],0;.ch_road);}
5390 '}
5400 'put_(x,y,num;ix);
5410 '{
5420 ' Q_put(x*2,y*2;ix);
5430 ' array(x,y;.IV);
5440 ' memory[ix]:=num;memory[ix,1]:=num;
5450 ' set ix:=ix+32;
5460 ' memory[ix]:=num;memory[ix,1]:=num;
```

```
5470 '}
5480 'Q_put(x,y;ix);
5490 '{
5500 ' put(x,y;ix);put(x+2,y;ix);
5510 ' put(x,y+2;ix);put(x+2,y+2;ix);
5520 '}
5530 'move_w(;IX,IY);
5540 '{
5550 ' memory[IX]:=memory[IY];memory[IX,1]:=memory[IY,1];
5560 '}
5570 'compare_w(;IX.IY);
5580 '{ ?(memory[IX,1]>memory[IY,1]:1,?(memory[IX]>=memory[IY]:1,0));}
5590 'sgn();
5600 '{
5610 'inline
5620 ' $ed,$5f,
5630 ' $e6,$01,
5640 ' $87,
5650 ' $3d;
5660 '}
5670 'sound(i;#,#);
5680 'var PORT40 at ($e6c1),on,off,j;
5690 '{
5700 ' off:=PORT40;on:=off or $20;inline $f3;
5710 ' if i=0
5720 '  then{port[$40]:=on;for i:=0 to 200{;}port[$40]:=off;}
5730 '  else{
5740 '   loop i,255{
5750 '    port[$40]:=on;for j:=0 to i{;};
5760 '    port[$40]:=off;for j:=0 to i{;};
5770 '   }
5780 '  }
5790 ' inline $fb;
5800 '}
5810 'var PRTFLG at($e64c);data _lpt:00;print_str(len;ix,#);{PRTFLG:=_lpt;inline
$dd,$e5,$e1,$7e,$b7,$28,$04,$df,$23,$18,$f8;PRTFLG:=0;}
5820 'print_num(num_l,num_h;#,#);{PRTFLG:=_lpt;inline$2a,#.num_l,$cd,#$28c2;PRTF
LG:=0;}
5830 'print_using(len,low_num,high_num;#,#);{PRTFLG:=_lpt;inline$2a,#.low_num,$c
d,$fd,$21,$3a,#.len,$47,$3e,$80,$0e,$00,$cd,$d1,$28,$cd,#$5550;PRTFLG:=0;}
5840 'input_chr(;#,#);{inline$cd,#$3583;}
5850 'data _mul:00,_mul_w:00,00;mul(a,b;#,#);var mul_m at(_code+6);{inline$21,#.
b,$3a,#.a,$cd,#.mul_m,$af,$94,$7c,$32,#._mul,$7d;}
5860 'mulov(;#,#);{_mul;}
5870 'mul_w(a_l,a_h,b_l,b_h;ix,#);{inline$c5,$21,#0000,$ed,$4b,#.a_l,$ed,$5b,#.b
_l,$3e,$10,$29,$cb,$11,$cb,$10,$30,$04,$19,$30,$01,$03,$3d,$20,$f2,$ed,$43,#._mu
l_w,$dd,$75,$00,$dd,$74,$01,$90,$3e,$00,$99,$c1;}
5880 'mulov_w(;ix,#);{@[ix,1]:=_mul_w[1];@[ix]:=_mul_w;}
5890 'array(x,y,z;#ix,#);var adr_l,adr_h;{if @[ix,4]-1 then{adr_l:=mul(z,@[ix,3]
);adr_h:=mulov();inline$2a,#.adr_l,$3a,#.y,$5f,$16,$00,$19,$22,#.adr_l;mul_w(adr
_l,adr_h,@[ix,2],0;.adr_l);}else{adr_l:=mul(y,@[ix,2]);adr_h:=mulov();
5900 '}inline$2a,#.adr_l,$dd,$5e,00,$dd,$56,01,$19,$3a,#.x,$5f,$16,$00,$19,$e5,$
dd,$e1;}
5910 'load_array(x,y,z;ix,#);{array(x,y,z;ix);@[ix];}
5920 'store_array(a,x,y,z;ix,#);{array(x,y,z;ix);@[ix]:=a;}
5930 '
```

3-6 コンパイラのダンプ・リストと打ち込みかた

ＰＣ-8801バージョンのStellarコンパイラの全ダンプ・リストを**リスト3-10**に示します。このとおりに打ち込めば，すぐにStellarコンパイラが使えるようになります。リストのチェックサムは8バイトのデータの和です。このプログラムの実行アドレスは&ＨＢ500で，ファイル名は“Stelar. b5”としてセーブしてください。

●リスト3-10　　ダンプ・リスト

```
B500 F3 01 00 34 2A 58 E6 7C :0C    B5F0 D4 3E 18 10 E3 ED 73 72 :EF
B508 B7 20 44 3A C2 E6 CB CF :97    B5F8 F3 31 F0 F2 F5 3A C2 E6 :DD
B510 D3 31 5E 23 56 7A B3 28 :30    B600 D3 31 F1 E9 F3 F5 3A C2 :C2
B518 0A EB E5 09 EB 72 2B 73 :DE    B608 E6 CB CF D3 31 F1 31 00 :A6
B520 E1 18 EF 5D 54 09 CB 7C :E9    B610 00 E1 C9 F3 F5 3A C2 E6 :74
B528 28 09 3A C2 E6 D3 31 FB :12    B618 CB CF D3 31 F1 CD 00 01 :5D
B530 C3 AB 03 EB D5 4D 44 03 :C5    B620 3A C2 E6 D3 31 FB C3 88 :2C
B538 ED B8 D1 3A C2 E6 D3 31 :5C    B628 60 F3 3A C2 E6 CB CF D3 :A2
B540 21 01 34 22 58 E6 13 ED :B6    B630 31 C3 6A 01 F5 3A C2 E6 :36
B548 53 18 EB AF 32 1A EB 01 :3D    B638 D3 31 F1 FB C9 3A C2 E6 :9B
B550 9C 00 11 20 F3 21 BD B5 :53    B640 D3 31 FB E9 F3 3A C2 E6 :BD
B558 ED B0 01 00 2F 11 00 01 :DF    B648 CB CF D3 31 C3 AB 04 3A :4A
B560 21 58 B6 ED B0 21 5D 07 :51    B650 00 EB B7 C8 3E 27 DF C9 :77
B568 36 00 11 5E 07 01 97 00 :44    B658 FE 57 28 37 FE 52 28 04 :30
B570 ED B0 AF 32 FF F2 01 03 :73    B660 F1 C3 97 F3 21 E5 07 11 :5C
B578 00 11 B6 EE 21 A2 B5 ED :1A    B668 16 F2 06 04 4E 23 7E 23 :24
B580 B0 01 15 00 11 81 E6 21 :5F    B670 12 1B 79 12 1B 10 F5 11 :E9
B588 A8 B5 ED B0 01 03 00 11 :0F    B678 06 F2 06 02 4E 23 7E 23 :12
B590 18 ED 21 A5 B5 ED B0 21 :3E    B680 12 1B 79 12 1B 10 F5 01 :D9
B598 00 00 22 62 07 22 01 00 :AE    B688 04 00 11 FD F1 21 F1 07 :1C
B5A0 FB C9 C3 8C F3 C3 B2 F3 :6E    B690 ED B0 C9 21 16 F2 11 E5 :85
B5A8 E3 F5 3E 60 BC 28 03 F1 :4E    B698 07 06 04 4E 2B 7E 2B 12 :45
B5B0 E3 C9 F1 E3 FE 5A C0 CD :65    B6A0 13 79 12 13 10 F5 21 06 :DD
B5B8 DD 6F C3 76 F3 F3 F5 3A :9A    B6A8 F2 06 02 4E 2B 7E 2B 12 :2E
B5C0 C2 E6 CB CF D3 31 F1 C3 :FA    B6B0 13 79 12 13 10 F5 01 04 :BB
B5C8 A6 05 F3 F5 3A C2 E6 CB :40    B6B8 00 21 FD F1 11 F1 07 ED :05
B5D0 CF D3 31 F1 C3 DD 05 F3 :5C    B6C0 B0 C9 AF 32 FF F2 7E FE :C7
B5D8 F5 3A C2 E6 CB CF D3 31 :75    B6C8 43 CA 90 01 FE C1 CA BC :E3
B5E0 F1 C3 16 06 CD 57 F3 CD :B4    B6D0 01 FE 8A CA 74 04 FE 99 :62
B5E8 0D 3E 18 18 CD 57 F3 CD :5F    B6D8 CA D5 04 FE 93 CA 07 05 :0A
```

```
B6E0 FE 9C CA 9E 05 C3 97 F3 :54
B6E8 23 7E FE 4F C2 97 F3 23 :5D
B6F0 7E FE 4D C2 97 F3 23 7E :B6
B6F8 FE 50 C2 97 F3 CD B3 06 :20
B700 C4 89 06 CD 46 07 E5 21 :73
B708 00 00 22 64 07 CD 00 08 :62
B710 E1 C3 97 F3 CD B3 06 C4 :78
B718 89 06 CD 46 07 E5 CD CD :28
B720 01 E1 C3 97 F3 ED 73 5D :EC
B728 07 31 E5 07 21 03 02 CD :17
B730 2E 07 21 00 C0 22 66 07 :A5
B738 21 00 00 22 64 07 CD 61 :DC
B740 04 FE 55 28 0B FE 65 C2 :AF
B748 30 04 21 22 02 CD 2E 07 :7B
B750 CD 61 04 D6 80 C2 30 04 :7E
B758 C3 29 03 0D 0A 4C 6F 61 :22
B760 64 20 73 74 65 6C 6C 61 :09
B768 72 20 63 6F 6D 70 69 6C :16
B770 65 72 20 6F 62 6A 65 63 :FA
B778 74 00 0D 0A 2A 2A 20 20 :1F
B780 44 65 62 75 67 20 6D 6F :E3
B788 64 65 20 6F 62 6A 65 63 :EC
B790 74 20 28 20 6E 6F 74 20 :4D
B798 42 53 41 56 45 20 29 00 :BA
B7A0 DB 09 07 38 06 DB 09 CB :D8
B7A8 77 20 FA CD 61 04 FE FF :C0
B7B0 CA 00 04 FE 56 DA 25 03 :24
B7B8 FE 5B D2 25 03 D6 56 87 :06
B7C0 6F 26 00 11 90 02 19 5E :AF
B7C8 23 56 EB CD 2E 07 CD 61 :94
B7D0 04 B7 28 05 CD 37 07 18 :0B
B7D8 F5 21 21 03 CD 2E 07 2A :66
B7E0 60 07 CD FF 06 C3 48 02 :46
B7E8 9A 02 AE 02 C1 02 D4 02 :E5
B7F0 E7 02 0D 0A 0A 50 72 6F :3B
B7F8 67 72 61 6D 20 20 6E 61 :B6
B800 6D 65 20 3A 20 00 0D 0A :63
B808 46 75 6E 63 74 69 6F 6E :46
B810 20 6E 61 6D 65 20 3A 20 :3B
B818 00 0D 0A 43 6F 6E 73 74 :1E
B820 61 6E 74 20 6E 61 6D 65 :04
B828 20 3A 20 00 0D 0A 56 61 :48
B830 72 69 61 62 6C 65 20 6E :FD
B838 61 6D 65 20 3A 20 00 0D :BA
B840 0A 44 61 74 61 20 20 20 :E4
B848 20 20 6E 61 6D 65 20 3A :3B
B850 20 00 0D 0A 0A 45 6E 64 :58
B858 20 61 64 64 72 65 73 73 :06
B860 20 20 20 3A 20 00 0D 0A :D1
B868 50 72 6F 67 72 61 6D 20 :F8
B870 73 69 7A 65 20 20 3A 20 :55
B878 00 20 3D 20 00 FE 80 20 :1B
B880 18 F5 CD 61 04 E5 CD 61 :52
B888 04 E1 67 22 60 07 F1 B7 :7D
B890 C2 48 02 22 62 07 C3 48 :A2
B898 02 FE 81 20 1E CD 61 04 :F1
B8A0 E5 CD 61 04 D1 57 ED 4B :77
B8A8 60 07 EB CD EC 03 5E 23 :8F
B8B0 56 70 2B 71 7A B3 20 F2 :A1
B8B8 C3 48 02 FE 66 20 1E 3E :ED
B8C0 CD CD DF 03 3E 20 CD DF :86
B8C8 03 3E F3 CD DF 03 CD 61 :11
B8D0 04 CD DF 03 CD 61 04 CD :B2
B8D8 DF 03 C3 48 02 FE 67 20 :74
B8E0 1D 3E CD CD DF 03 3E 2D :42
B8E8 CD DF 03 3E F3 CD DF 03 :8F
B8F0 CD 61 04 F5 CD DF 03 F1 :C7
B8F8 B7 20 F5 C3 48 02 FE 68 :3F
B900 20 1E 3E CD CD DF 03 3E :36
B908 3A CD DF 03 3E F3 CD DF :C6
B910 03 CD 61 04 CD DF 03 CD :B1
B918 61 04 CD DF 03 C3 48 02 :21
B920 FE 41 D2 30 04 47 B7 20 :63
B928 01 04 C5 CD 61 04 CD DF :A8
B930 03 C1 10 F6 C3 48 02 2A :01
B938 60 07 23 22 60 07 2B CD :0B
B940 EC 03 77 C9 EB 2A 54 E6 :7E
B948 2B B7 ED 52 30 4F 21 FF :C0
B950 E5 B7 ED 52 38 47 EB C9 :0E
B958 21 FA 02 CD 2E 07 2A 60 :A9
B960 07 2B CD FF 06 21 0E 03 :36
B968 CD 2E 07 2A 60 07 ED 5B :DB
B970 62 07 B7 ED 52 CD FF 06 :31
B978 2A 62 07 18 03 21 00 00 :CF
B980 22 64 07 ED 7B 5D 07 C9 :22
B988 21 38 04 CD 2E 07 18 ED :64
B990 0D 0A 42 61 64 20 6F 62 :0F
B998 6A 65 63 74 00 D5 CD 25 :6D
B9A0 07 E1 CD FF 06 21 55 04 :34
B9A8 CD 2E 07 18 D0 20 4C 6F :C5
B9B0 61 64 20 65 72 72 6F 72 :0F
B9B8 00 2A 66 07 7C B7 CA 30 :C4
B9C0 04 D3 5D 7E D3 5F 23 22 :29
B9C8 66 07 6F C9 CD B3 06 28 :53
B9D0 13 FE 0C C2 97 F3 23 5E :EA
B9D8 23 56 ED 53 64 07 CD B3 :A4
B9E0 06 C2 97 F3 EB 2A 64 07 :D2
B9E8 7C B5 20 04 EB C3 97 F3 :8D
B9F0 D5 ED 73 5D 07 31 FE F2 :BA
B9F8 AF 32 5F 07 11 A7 F3 D5 :C7
BA00 C3 A0 F3 AF 32 5F 07 21 :BE
BA08 BD 04 CD 2E 07 ED 7B 5D :88
BA10 07 E1 C3 97 F3 0D 0A 2A :76
BA18 2A 20 45 6E 64 20 6F 66 :56
BA20 20 65 78 65 63 75 74 69 :17
BA28 6F 6E 0D 0A 00 CD B3 06 :7A
BA30 C2 97 F3 3A 5F 07 B7 CA :6D
BA38 97 F3 E5 ED 73 5D 07 31 :64
BA40 E5 07 ED 5B F3 07 2A F1 :49
BA48 07 2B 72 2B 73 22 F1 07 :5C
BA50 F1 C1 D1 E1 DD E1 FD E1 :00
BA58 ED 7B F1 07 C3 97 F3 CD :7A
```

```
BA60 B3 06 CA 97 F3 06 22 B8 :ED
BA68 C2 97 F3 0E 00 23 22 6C :0B
BA70 05 7E B7 20 03 2B 18 07 :A7
BA78 B8 28 04 0C 23 18 F2 CD :EA
BA80 B3 06 C2 97 F3 22 96 05 :C2
BA88 79 B7 CA 97 F3 32 6A 05 :25
BA90 2A 58 E6 DB 09 07 38 0A :95
BA98 DB 09 0F D2 95 05 CB 6F :99
BAA0 20 F6 5E 23 56 23 7A B3 :3D
BAA8 CA 95 05 ED 53 68 07 5E :71
BAB0 23 56 23 EB 22 AB E6 22 :5C
BAB8 BE EF 13 13 13 ED 53 87 :AD
BAC0 05 0E 00 21 00 00 41 1A :8F
BAC8 B7 28 1C BE 13 20 F4 23 :03
BAD0 10 F5 2A AB E6 CD C8 06 :5B
BAD8 21 9B 05 CD 2E 07 21 00 :E4
BAE0 00 CD 2E 07 CD 25 07 2A :25
BAE8 68 07 C3 3B 05 21 00 00 :93
BAF0 C3 97 F3 20 27 00 3E FF :D1
BAF8 32 FF F2 C3 07 05 ED 73 :52
BB00 F1 07 31 F1 07 FD E5 DD :E0
BB08 E5 E5 D5 C5 F5 2A F1 07 :7B
BB10 5E 23 56 23 22 F1 07 ED :01
BB18 53 F3 07 3E 5B CD 47 F3 :ED
BB20 2A F3 07 5E 23 56 23 22 :40
BB28 F3 07 EB CD C8 06 3E 5D :1B
BB30 CD 47 F3 18 72 ED 73 F1 :E2
BB38 07 31 F1 07 FD E5 DD E5 :D4
BB40 E5 D5 C5 F5 2A F1 07 5E :F4
BB48 23 56 23 22 F1 07 ED 53 :F6
BB50 F3 07 3E 7B CD 47 F3 2A :E4
BB58 F3 07 7E 23 B7 28 05 CD :4C
BB60 47 F3 18 F6 22 F3 07 3E :A2
BB68 7D CD 47 F3 18 39 ED 73 :35
BB70 F1 07 31 F1 07 FD E5 DD :E0
BB78 E5 E5 D5 C5 F5 2A F1 07 :7B
BB80 5E 23 56 23 22 F1 07 ED :01
BB88 53 F3 07 21 79 06 CD 2E :E8
BB90 07 21 84 06 CD 2E 07 2A :DE
BB98 F3 07 5E 23 56 23 22 F3 :09
BBA0 07 EB CD C8 06 18 1D DB :9D
BBA8 09 07 38 0B DB 09 CB 77 :79
BBB0 28 09 0F 30 09 18 F5 0F :95
BBB8 0F 30 03 C3 E7 04 21 79 :8A
BBC0 06 CD 2E 07 3E FF 32 5F :D6
BBC8 07 ED 7B 5D 07 E1 C3 97 :0E
BBD0 F3 0D 0A 2A 2A 20 42 72 :32
BBD8 65 61 6B 00 20 69 6E 20 :48
BBE0 00 7E FE 22 20 21 CD BF :6B
BBE8 06 C8 FE 22 28 15 FE 70 :99
BBF0 28 04 FE 50 20 11 3E FF :E8
BBF8 32 FF F2 CD BF 06 C8 FE :7B
BC00 22 20 04 CD B3 06 C8 F1 :85
BC08 C3 97 F3 23 7E B7 C8 FE :6B
BC10 3A C8 FE 20 C0 18 F4 23 :0F
BC18 7E B7 C8 FE 20 C0 18 F7 :EA
BC20 16 00 01 10 27 CD E8 06 :09
BC28 01 E8 03 CD E8 06 01 64 :0C
BC30 00 CD E8 06 01 0A 00 CD :93
BC38 E8 06 7D F6 30 C3 37 07 :92
BC40 1E 30 B7 ED 42 1C 30 FA :7A
BC48 09 1D 7B FE 30 20 03 15 :07
BC50 14 C8 14 7B C3 37 07 E5 :51
BC58 7C CD 06 07 E1 7D F5 0F :B8
BC60 0F 0F 0F CD 0F 07 F1 E6 :E7
BC68 0F C6 30 FE 3A 38 02 C6 :3D
BC70 07 18 1C 3E 20 18 18 CD :96
BC78 1B 07 10 FB C9 3E 0D CD :0E
BC80 37 07 3E 0A 18 09 7E B7 :DC
BC88 C8 CD 37 07 23 18 F7 D5 :DA
BC90 5F 3A FF F2 B7 7B D1 C4 :51
BC98 4F F3 CD 47 F3 C9 3A C2 :0E
BCA0 E6 CB 9F 32 C2 E6 CB CF :C4
BCA8 D3 31 3A C1 E6 CB E7 32 :C9
BCB0 C1 E6 D3 40 C9 3B C8 D1 :57
BCB8 C3 9D 04 06 00 C5 E5 2B :3F
BCC0 CD C8 0B FE 27 CA 7D 33 :3F
BCC8 FE 22 CA 7D 33 FE 3C CA :9E
BCD0 9B 33 FE 3E CA A6 33 FE :AB
BCD8 25 CA D3 32 FE 2C CA B0 :98
BCE0 33 FE 20 CA B0 33 FE 09 :05
BCE8 CA B0 33 FE 3B CA B0 33 :93
BCF0 FE 21 CC 29 0B FE 0D CA :F4
BCF8 CD 33 77 2B CD EA 0A C3 :26
BD00 B5 22 51 2F 21 00 D5 22 :6F
BD08 53 2F 21 00 E6 22 55 2F :2F
BD10 CD A2 0D 21 05 09 CD B1 :29
BD18 0A 11 08 09 ED 4B 57 2F :EA
BD20 2A 51 2F CD 61 09 11 14 :06
BD28 09 ED 4B 59 2F 2A 53 2F :75
BD30 CD 61 09 21 20 09 CD B1 :FF
BD38 0A 2A 55 2F 2B CD E9 09 :A2
BD40 21 37 09 CD B1 0A 21 39 :43
BD48 09 CD B1 0A 2A 3C 2F 7C :A2
BD50 B5 20 08 21 52 09 CD B1 :D7
BD58 D3 5C 21 39 2F 36 00 11 :FF
BD60 3A 2F 01 A0 00 ED B0 ED :94
BD68 73 39 2F 31 39 33 21 00 :99
BD70 C0 22 3E 2F 01 00 40 D3 :63
BD78 5D 36 FF 23 0B 78 B1 20 :09
BD80 F8 D3 5C 21 AC 08 CD B1 :7A
BD88 0A 21 00 00 22 3C 2F D3 :8B
BD90 5F 2A 58 E6 D3 5C 22 42 :5A
BD98 2F 21 00 B5 22 51 2F 21 :C8
BDA0 00 D5 22 53 2F 21 00 E6 :80
BDA8 22 55 2F CD A2 0D 21 05 :48
BDB0 09 CD B1 0A 11 08 09 ED :A0
BDB8 4B 57 2F 2A 51 2F CD 61 :A9
BDC0 09 11 14 09 ED 4B 59 2F :F7
BDC8 2A 53 2F CD 61 09 21 20 :24
BDD0 09 CD B1 0A 2A 55 2F 2B :6A
BDD8 CD E9 09 21 37 09 CD B1 :9E
```

```
BDE0 0A 21 39 09 CD B1 0A 2A :1F
BDE8 3C 2F 7C B5 20 08 21 52 :37
BDF0 09 CD B1 0A 18 05 3E 03 :EF
BDF8 CD 06 0A 21 55 09 CD B1 :DA
BE00 0A C3 D9 09 0D 0A 53 74 :8D
BE08 65 6C 6C 61 72 20 63 6F :02
BE10 6D 70 69 6C 65 72 20 52 :FB
BE18 65 76 20 31 2E 30 30 0D :C7
BE20 0A 20 28 20 50 43 2D 38 :6A
BE28 38 30 31 2F 6D 6B 49 49 :32
BE30 20 56 65 72 73 69 6F 6E :06
BE38 20 29 0D 0A 43 6F 70 79 :FB
BE40 72 69 67 68 74 20 28 63 :C9
BE48 29 20 31 39 38 34 20 48 :87
BE50 2E 4F 68 6E 75 6B 69 20 :BC
BE58 2F 20 4D 49 41 0D 0A 00 :3D
BE60 0D 0A 50 72 6F 67 72 61 :82
BE68 6D 20 20 00 0D 0A 44 61 :69
BE70 74 61 20 20 20 20 20 00 :75
BE78 0D 0A 53 74 61 63 6B 20 :2D
BE80 62 6F 74 74 6F 6D 20 20 :D5
BE88 28 20 20 20 20 2D 00 29 :FE
BE90 00 0D 0A 0A 2A 2A 20 20 :B5
BE98 45 6E 64 20 6F 66 20 63 :8F
BEA0 6F 6D 70 69 6C 65 2C 20 :D2
BEA8 20 00 4E 6F 00 20 65 72 :D4
BEB0 72 6F 72 28 73 29 0D 0A :2E
BEB8 00 C5 E5 EB CD B1 0A D1 :EE
BEC0 E1 E5 D5 B7 ED 52 7C B5 :C2
BEC8 F5 CD E9 09 F1 28 1D 3E :28
BED0 20 CD BA 0A 3E 28 CD BA :9E
BED8 0A E1 CD E9 09 3E 2D CD :E2
BEE0 BA 0A E1 2B CD E9 09 3E :CD
BEE8 29 C3 BA 0A F1 F1 C9 21 :7C
BEF0 9D 09 C3 D0 09 0D 0A 25 :7E
BEF8 4F 62 6A 65 63 74 20 61 :D8
BF00 72 65 61 20 66 75 6C 6C :0B
BF08 00 21 B7 09 C3 D0 09 0D :8A
BF10 0A 25 53 79 6D 62 6F 6C :A5
BF18 20 74 61 62 6C 65 20 6F :B7
BF20 76 65 72 66 6C 6F 77 00 :05
BF28 CD B1 0A 21 05 09 CD B1 :35
BF30 0A ED 7B 39 2F D3 5F 2A :36
BF38 4D 2F 22 AB E6 22 BE EF :FE
BF40 C9 E5 7C CD F0 09 E1 7D :4E
BF48 F5 0F 0F 0F 0F CD F9 09 :00
BF50 F1 E6 0F C6 30 FE 3A 38 :4C
BF58 02 C6 07 C3 BA 0A 57 01 :AE
BF60 10 27 CD 25 0A 01 E8 03 :1F
BF68 CD 25 0A 01 64 00 CD 25 :53
BF70 0A 01 0A 00 CD 25 0A 7D :8E
BF78 F6 30 C3 BA 0A 1E 30 B7 :B2
BF80 ED 42 1C 30 FA 09 1D CB :66
BF88 4A 28 0D 7B FE 30 20 06 :4E
BF90 CB 42 C0 1E 20 01 CB 8A :61
BF98 7B C3 BA 0A 21 4D 0A CD :47
```

```
BFA0 B1 0A C3 89 08 0D 0A 25 :4B
BFA8 41 62 6F 72 74 00 ED 53 :38
BFB0 48 2F 2B E5 21 05 09 CD :83
BFB8 B1 0A E1 23 7E B7 28 39 :55
BFC0 E5 FE 23 20 13 2A 4D 2F :DF
BFC8 3E 02 CD 06 0A 21 7D 0A :C5
BFD0 CD B1 0A 18 E5 3A 20 00 :DF
BFD8 FE 40 20 0C E1 23 5E 23 :EF
BFE0 56 E5 EB CD B1 0A 18 D2 :98
BFE8 FE 5C 20 08 2A 48 2F CD :F0
BFF0 B1 0A 18 C6 CD BA 0A 18 :42
BFF8 C1 2A 3C 2F 23 22 3C 2F :06
C000 21 05 09 CD B1 0A 2A 40 :21
C008 2F 7E B7 C8 CD BA 0A 23 :E0
C010 18 F7 5F D3 5F 3A FF F2 :CB
C018 B7 7B C4 4F F3 CD 47 F3 :3F
C020 D3 5C C9 21 D4 0A CD 56 :1A
C028 0A C3 44 0A 23 4E 6F 6E :69
C030 20 73 75 70 70 6F 72 74 :3D
C038 69 6E 67 20 3A 20 25 69 :46
C040 6E 63 6C 75 64 65 00 B7 :32
C048 C9 3A 4F 2F CB 67 C8 3E :B9
C050 68 CD 01 0B 2A 4D 2F 18 :FF
C058 30 CD BB 0B 2A 57 2F 23 :96
C060 23 23 22 57 2F C9 3A 4F :40
C068 2F CB 67 C8 3E FF 32 3B :D3
C070 2F C9 3A 4F 2F CB 67 C8 :AA
C078 AF 32 3B 2F C9 3A 3B 2F :B8
C080 B7 C8 E5 3E 66 CD 01 0B :E1
C088 E1 E5 7D CD BB 0B E1 7C :33
C090 CD BB 0B 2A 57 2F 23 23 :89
C098 22 57 2F C9 3A 3B 2F B7 :CC
C0A0 C8 E5 3E 67 CD 01 0B E1 :0C
C0A8 7E E5 CD BB 0B 2A 57 2F :A6
C0B0 23 22 57 2F E1 7E 23 B7 :04
C0B8 20 EE C9 2A 42 2F 5E 23 :F3
C0C0 56 23 ED 53 42 2F 7A B3 :57
C0C8 28 24 5E 23 56 D5 11 98 :A1
C0D0 0B 01 20 03 23 7E B7 CA :51
C0D8 9B 0B B9 28 F7 0E 00 1A :A6
C0E0 BE C2 9B 0B 13 23 10 ED :59
C0E8 D1 22 40 2F B7 C9 37 C9 :E2
C0F0 3A 8F E9 21 05 09 CD B1 :5F
C0F8 0A E1 3E 02 CD 06 0A 21 :29
C100 AD 0B C3 D0 09 3A 20 25 :D3
C108 42 61 64 20 73 6F 75 72 :F0
C110 63 65 00 5F 2A 3E 2F 7C :3A
C118 B5 CA 97 09 7B 23 22 3E :1D
C120 2F 2B D3 5D 77 D3 5C C9 :F9
C128 CB 66 28 2A 11 F0 FF 01 :84
C130 00 0C E5 D5 2A 46 2F B7 :1C
C138 ED 52 28 44 D1 E1 E5 D5 :17
C140 23 13 1A BE 20 07 B7 28 :14
C148 5E 10 F5 18 5A D1 21 F0 :B7
C150 FF 19 EB E1 18 D9 11 00 :E6
C158 C0 01 00 0C E5 D5 2A 44 :F5
```

```
C160 2F B7 ED 52 28 26 D1 E1 :25
C168 E5 D5 23 13 1A BE 20 07 :EF
C170 B7 28 34 10 F5 18 30 D1 :31
C178 21 10 00 19 EB E1 18 D9 :07
C180 2A 46 2F 11 F0 FF 19 22 :DA
C188 46 2F 18 0A 2A 44 2F 11 :45
C190 10 00 19 22 44 2F 2A 46 :2E
C198 2F 11 20 00 19 ED 5B 44 :05
C1A0 2F B7 ED 52 CA B1 09 D1 :7A
C1A8 E1 01 10 00 ED B0 C9 11 :69
C1B0 F0 FF 01 00 0C E5 D5 1A :D0
C1B8 B7 28 14 23 13 1A BE 20 :21
C1C0 13 B7 28 02 10 F5 E1 D1 :AB
C1C8 01 10 00 ED B0 AF C9 E1 :07
C1D0 2A 44 2F FE E1 11 F0 FF :7C
C1D8 19 E5 11 00 C0 B7 ED 52 :C5
C1E0 D1 E1 30 CE F6 FF C9 ED :5B
C1E8 5B 46 2F 21 F0 FF 22 46 :48
C1F0 2F B7 ED 52 44 4D 21 10 :E7
C1F8 00 19 16 00 1E 01 78 B1 :77
C200 C8 1D 20 25 1E 10 CB 7E :A1
C208 28 1F C5 D5 E5 23 E5 11 :DF
C210 0C 00 19 36 00 21 D6 0C :5E
C218 CD B1 0A E1 CD B1 0A 2A :1B
C220 3C 2F 23 22 3C 2F E1 D1 :CD
C228 C1 72 23 0B 18 D0 0D 0A :60
C230 20 20 20 20 3F 3A 20 55 :6E
C238 6E 64 65 66 69 6E 65 64 :3D
C240 20 6C 61 62 65 6C 20 3A :7A
C248 20 00 21 00 C0 22 44 2F :96
C250 01 00 40 36 00 23 0B 78 :1D
C258 B1 20 F8 21 F0 FF 22 46 :41
C260 2F C9 11 00 C0 2A 44 2F :66
C268 B7 ED 52 D5 DD E1 7C B5 :BA
C270 28 5E E5 DD 6E 0D DD 66 :06
C278 0E DD 36 0D 00 DD CB 00 :D6
C280 7E 28 19 DD E5 21 7E 0D :2D
C288 CD B1 0A E1 E5 23 CD B1 :EF
C290 0A 2A 3C 2F 23 22 3C 2F :4F
C298 DD E1 18 29 06 58 DD 7E :B8
C2A0 00 E6 0F 3D 28 08 04 3D :A3
C2A8 28 04 04 3D 20 17 C5 DD :46
C2B0 E5 CD 54 27 E1 F1 E5 23 :07
C2B8 11 71 2F 01 0D 00 ED B0 :5C
C2C0 CD 41 27 DD E1 11 10 00 :14
C2C8 DD 19 E1 B7 ED 52 18 9E :83
C2D0 2A 57 2F C3 54 27 0D 0A :05
C2D8 20 20 20 20 3F 3A 20 55 :6E
C2E0 6E 64 65 66 69 6E 65 64 :3D
C2E8 20 66 75 6E 63 74 69 6F :18
C2F0 6E 20 6E 61 6D 65 20 3A :89
C2F8 20 00 CD 63 0B D8 22 4B :A0
C300 2F ED 53 4D 2F CD B1 28 :91
C308 FE 80 C2 B0 2D CD B1 28 :C3
C310 3D C2 B0 2D CD 2E 27 AF :AD
C318 32 4F 2F CD B1 28 FE 28 :7C
C320 C2 B0 2D CD B1 28 FE 29 :6C
C328 CA 4F 0E FE 25 C2 60 2C :98
C330 CD B1 28 FE 84 20 0A 21 :73
C338 4F 2F CB E6 CD B1 28 18 :ED
C340 5A 3D C2 60 2C 2A 62 2F :A0
C348 7C B7 C2 60 2C 7D CD 57 :22
C350 2B F5 CD B1 28 FE 3A C2 :C0
C358 60 2C F1 FE 50 20 09 21 :15
C360 4F 2F CB FE 06 80 18 19 :FE
C368 FE 44 20 09 21 4F 2F CB :D5
C370 F6 06 40 18 0C FE 53 C2 :73
C378 60 2C 21 4F 2F CB EE 06 :EA
C380 20 C5 CD B1 28 CD A3 27 :22
C388 F1 87 30 03 22 51 2F 87 :D4
C390 30 03 22 53 2F 87 30 03 :91
C398 22 55 2F 3A 81 2F FE 2C :BA
C3A0 28 81 FE 29 C2 60 2C CD :EB
C3A8 B1 28 FE 3B C2 B0 2D 3A :EB
C3B0 4F 2F CB 67 3E 55 28 02 :6D
C3B8 3E 65 CD BB 0B 2A 51 2F :E0
C3C0 22 57 2F CD 54 27 2A 53 :6D
C3C8 2F 22 59 2F 22 5B 2F 3E :C3
C3D0 56 CD 41 27 21 93 11 CD :1D
C3D8 B1 0A 21 71 2F CD B1 0A :04
C3E0 CD B9 11 CD F2 0C 2A 53 :DF
C3E8 2F CD 29 12 5F 77 6F 72 :EE
C3F0 6B 00 2A 59 2F CD 29 12 :25
C3F8 5F 76 61 72 00 2A 51 2F :52
C400 CD 29 12 5F 63 6F 64 65 :02
C408 00 16 3E 1E 01 CD 06 27 :6D
C410 3E 32 2A 53 2F 11 21 00 :4E
C418 19 CD 10 27 3E 01 B7 28 :3B
C420 22 3D 20 08 11 73 ED CD :C5
C428 06 27 18 0D 3E 21 21 00 :D2
C430 00 CD 10 27 3E 22 CD F5 :26
C438 26 2A 53 2F 11 22 00 19 :1E
C440 CD 08 27 3A 4F 2F 2A 55 :33
C448 2F CB 6F 20 15 3E 00 21 :FD
C450 00 E6 B7 28 0D E5 11 7B :43
C458 ED CD 06 27 E1 CD 08 27 :C4
C460 18 05 3E 31 CD 10 27 3E :CE
C468 21 2A 51 2F 11 03 00 19 :F8
C470 CD 10 27 3E E5 CD F5 26 :0F
C478 CD B1 28 AF 32 D7 2F CD :5A
C480 40 12 FE 7B C2 B0 2D 3E :A8
C488 FF 32 50 2F 2A 4D 2F CD :23
C490 25 0B CD E7 18 CD 2A 15 :08
C498 CD FF 18 AF 32 50 2F CD :11
C4A0 B1 28 3E C9 CD F5 26 CD :95
C4A8 8F 0C 3A 81 2F 3C CA 8A :15
C4B0 11 AF 32 D7 2F 67 6F 22 :F0
C4B8 D8 2F F5 CD 40 12 2F 32 :7C
C4C0 82 2F 3A 81 2F FE FF 20 :B8
C4C8 04 F1 C3 8A 11 FE 8F 20 :00
C4D0 07 F1 F6 80 F5 CD B1 28 :09
C4D8 3D C2 79 2C 21 61 2F CD :22
```

```
C4E0 57 0C 06 09 B7 20 16 3A :99
C4E8 61 2F FE 89 28 07 C5 CD :D8
C4F0 96 2C C1 18 08 2A 6E 2F :6A
C4F8 F5 CD 59 27 C1 2A 57 2F :B3
C500 22 6E 2F CB B8 21 61 2F :F3
C508 70 CD D0 0B CD 2E 27 3E :78
C510 57 CD 41 27 21 A6 11 CD :31
C518 B1 0A 21 71 2F E5 CD B1 :DF
C520 0A E1 CD 44 0B CD B1 28 :AD
C528 FE 28 C2 B0 2D 2A 59 2F :77
C530 22 D3 2F AF 32 D5 2F CD :D6
C538 B1 28 FE 3B 28 44 FE 29 :A5
C540 28 7B 3D C2 A8 2C 18 03 :91
C548 CD B1 28 CD 04 14 F6 02 :83
C550 2A 59 2F 22 6E 2F 21 61 :F3
C558 2F 77 CD D0 0B 2A 59 2F :00
C560 23 22 59 2F 22 5B 2F 21 :9A
C568 D5 2F 7E FE 20 30 01 34 :05
C570 CD B1 28 FE 2C 28 D1 FE :C7
C578 3B 28 07 FE 29 28 3E C3 :BA
C580 B0 2D CD B1 28 FE 23 20 :C4
C588 07 F1 F6 08 F5 CD B1 28 :91
C590 FE BF 20 07 F1 F6 04 F5 :C4
C598 CD B1 28 FE 2C 20 19 CD :D6
C5A0 B1 28 FE 23 20 07 F1 F6 :08
C5A8 02 F5 CD B1 28 FE C0 20 :7B
C5B0 07 F1 F6 01 F5 CD B1 28 :8A
C5B8 FE 29 C2 B0 2D F1 CB 57 :D9
C5C0 28 0E F5 3E 22 2A 53 2F :37
C5C8 11 2C 00 19 CD 10 27 F1 :4B
C5D0 CB 47 28 12 F5 11 53 ED :92
C5D8 CD 06 27 2A 53 2F 11 2E :E5
C5E0 00 19 CD 08 27 F1 F5 3A :35
C5E8 D5 2F 5F 16 0E CD 06 27 :81
C5F0 3E 11 21 00 00 CD 10 27 :74
C5F8 2A D3 2F F1 CB 7F 28 08 :97
C600 E5 ED 5B 57 2F 1B 1B D5 :BE
C608 F5 3E 21 CD 10 27 3E CD :63
C610 2A 51 2F 11 24 00 19 CD :C5
C618 10 27 F1 CB 5F 20 0D F5 :74
C620 11 E5 DD CD 06 27 3E FF :0A
C628 32 D7 2F F1 CB 4F 20 0D :70
C630 F5 11 E5 FD CD 06 27 3E :20
C638 FF 32 D7 2F F1 CB 57 28 :72
C640 12 F5 11 2A DD CD 06 27 :19
C648 2A 53 2F 11 2C 00 19 CD :CF
C650 08 27 F1 CB 47 28 12 F5 :61
C658 11 2A FD CD 06 27 2A 53 :AF
C660 2F 11 2E 00 19 CD 08 27 :83
C668 F1 F5 CD B1 28 FE 3B C2 :87
C670 B0 2D CD B1 28 3E FF CD :8D
C678 40 12 FE 7B C2 B0 2D 3E :A8
C680 FF 32 50 2F 2A 4D 2F CD :23
C688 25 0B CD E7 18 CD 2A 15 :08
C690 CD FF 18 AF 32 50 2F CD :11
C698 B1 28 2A D8 2F 7C B5 C4 :FF
```

```
C6A0 59 27 F1 CB 4F 20 08 F5 :A8
C6A8 11 E1 FD CD 06 27 F1 CB :A5
C6B0 5F 20 08 F5 11 E1 DD CD :18
C6B8 06 27 F1 F5 3E C9 CD F5 :DC
C6C0 26 F1 87 D2 4F 0F E1 CD :7C
C6C8 54 27 D1 2A 59 2F B7 ED :A2
C6D0 52 CD 08 27 2A 57 2F 2B :29
C6D8 2B 22 57 2F CD 54 27 C3 :DE
C6E0 4F 0F CD 0A 0D 3E FF CD :4C
C6E8 BB 0B C9 0D 0A 50 72 6F :D7
C6F0 67 72 61 6D 20 20 6E 61 :B6
C6F8 6D 65 20 3A 20 00 0D 0A :63
C700 46 75 6E 63 74 69 6F 6E :46
C708 20 6E 61 6D 65 20 3A 20 :3B
C710 00 01 F7 00 21 FB 2D DD :1E
C718 21 F2 2E FD 21 14 2F 7D :1F
C720 DD BE 00 20 1B 7C DD BE :ED
C728 01 20 15 DD 23 DD 23 11 :47
C730 FB 2D 7E 93 5F 23 7E 9A :D3
C738 57 E5 2A 51 2F 19 18 29 :40
C740 7D FD BE 00 20 1B 7C FD :EC
C748 BE 01 20 15 FD 23 FD 23 :34
C750 11 DA 2F 7E 93 5F 23 7E :2B
C758 9A 57 E5 2A 53 2F 19 18 :B3
C760 08 7E E5 C5 CD F5 26 18 :30
C768 05 0B C5 CD 08 27 C1 E1 :73
C770 23 0B 78 B1 20 A9 2A 59 :A3
C778 2F 11 30 00 19 22 59 2F :33
C780 C9 22 6E 2F 21 61 2F 36 :6F
C788 01 D1 1A 13 23 77 B7 20 :70
C790 F9 D5 21 61 2F C3 D0 0B :1D
C798 32 82 2F 3A 81 2F FE 25 :F0
C7A0 20 08 CD B1 28 CD B7 14 :66
C7A8 18 F1 FE 8C 20 05 CD 6C :F1
C7B0 12 18 E8 FE 8D 20 05 CD :8F
C7B8 B1 12 18 DF FE 8E C0 CD :D3
C7C0 4E 13 18 D7 CD B1 28 3D :33
C7C8 C2 A8 2C CD 04 14 F6 01 :72
C7D0 01 0D 00 21 61 2F 77 CD :03
C7D8 95 28 CD B1 28 FE F0 C2 :13
C7E0 B0 2D CD B1 28 CD 69 27 :E0
C7E8 22 6E 2F 01 0D 00 21 61 :4F
C7F0 2F CD A5 28 21 61 2F CD :47
C7F8 D0 0B 3A 81 2F FE 2C 28 :17
C800 C3 FE 3B C2 B0 2D C3 B1 :0F
C808 28 CD B1 28 3D C2 A8 2C :A1
C810 CD 04 14 F6 02 2A 59 2F :8F
C818 22 D3 2F 21 01 00 22 D5 :3D
C820 2F 01 0D 00 21 61 2F 77 :65
C828 CD 95 28 CD B1 28 FE 5B :89
C830 20 14 CD B1 28 CD 69 27 :37
C838 22 D5 2F 3A 81 2F FE 5D :6B
C840 C2 B0 2D CD B1 28 FE BE :01
C848 20 22 CD B1 28 FE 28 C2 :D0
C850 B0 2D CD B1 28 CD 69 27 :E0
C858 22 D3 2F 21 00 00 22 D5 :3C
```

```
C860 2F 3A 81 2F FE 29 C2 B0 :B2
C868 2D CD B1 28 2A D3 2F 22 :21
C870 6E 2F 01 0D 00 21 61 2F :5C
C878 CD A5 28 21 61 2F CD D0 :E8
C880 0B 2A 59 2F ED 5B D5 2F :09
C888 19 22 59 2F 22 5B 2F 3A :A9
C890 81 2F FE 2C CA B1 12 FE :65
C898 3B C2 B0 2D C3 B1 28 3E :B4
C8A0 02 32 83 2F 18 18 3E 01 :55
C8A8 32 83 2F 3E C3 21 00 00 :06
C8B0 CD 10 27 2A 57 2F 2B 2B :0A
C8B8 22 CF 2F CD B1 28 3A 83 :83
C8C0 2F 3D 20 27 3A 81 2F 3D :DA
C8C8 20 21 2A 4B 2F 7E FE 3A :9B
C8D0 20 19 23 22 4B 2F CD 04 :C9
C8D8 14 F6 03 2A 57 2F 22 6E :4D
C8E0 2F 21 61 2F 77 CD D0 0B :FF
C8E8 CD B1 28 3A 81 2F FE 22 :B0
C8F0 20 2E 2A 4B 2F 7E 23 FE :91
C8F8 22 20 0F 7E 23 FE 22 28 :3A
C900 09 2B 22 4B 2F CD B1 28 :76
C908 18 35 B7 20 0C 2B 22 4B :C8
C910 2F CD EA 2C CD B1 28 18 :D0
C918 26 E5 CD F5 26 E1 18 D5 :C1
C920 FE 23 28 04 FE 92 20 0B :08
C928 CD B1 28 CD A3 27 CD 08 :12
C930 27 18 0C FE 91 CC B1 28 :7F
C938 CD 8F 27 7D CD F5 26 3A :22
C940 81 2F FE 2C CA 63 13 FE :18
C948 3B C2 B0 2D 21 83 2F 7E :2B
C950 36 00 3D 2A CF 2F CC 59 :C0
C958 27 C3 B1 28 21 61 2F CD :41
C960 57 0C B7 20 13 3A 82 2F :38
C968 B7 20 05 CD BC 2C 18 08 :B1
C970 3A 61 2F CB 67 C4 BC 2C :A8
C978 3A 82 2F B7 C8 3E 10 C9 :81
C980 3A 81 2F 3D 20 41 2A 4B :FD
C988 2F 7E FE 3A 20 39 23 7E :DF
C990 FE 3D 28 33 22 4B 2F 21 :53
C998 61 2F CD 57 0C B7 20 16 :AD
C9A0 3A 61 2F CB 67 28 0F FE :31
C9A8 98 28 05 CD CE 2C 18 06 :AA
C9B0 2A 6E 2F CD 59 27 2A 57 :95
C9B8 2F 22 6E 2F 21 61 2F 36 :D5
C9C0 18 CD D0 0B CD B1 28 CD :33
C9C8 D3 25 3A 81 2F 21 90 14 :A7
C9D0 01 0D 00 ED B1 C2 13 19 :9A
C9D8 3E 0C 91 87 4F 21 9D 14 :83
C9E0 09 5E 23 56 D5 C3 B1 28 :51
C9E8 3B 25 7B 90 96 98 9B 93 :C7
C9F0 9E 9D 9C 9F A0 29 15 B7 :0B
C9F8 14 FE 14 47 13 38 15 6E :3B
CA00 15 A4 16 99 17 1D 18 3D :F1
CA08 18 35 18 93 18 B8 18 FE :DE
CA10 81 20 06 CD CB 0A C3 B1 :BD
CA18 28 FE 8B 20 15 CD B1 28 :8C
CA20 FE 3B C2 B0 2D 3A 50 2F :91
CA28 B7 CA B0 2D CD F1 0A C3 :E9
CA30 B1 28 D6 85 20 0E CD B1 :E0
CA38 28 FE 3B C2 B0 2D CD 0E :DB
CA40 0B C3 B1 28 3D C2 B0 2D :83
CA48 CD B1 28 FE 3B C2 B0 2D :7E
CA50 CD 1A 0B C3 B1 28 CD E7 :42
CA58 18 CD 2D 15 CD B1 28 FE :CB
CA60 97 20 1B CD D3 25 CD B1 :15
CA68 28 CD 51 1B 3E B7 CD F5 :18
CA70 26 2A C5 2F 3E 28 CD CC :43
CA78 18 3E CA DC 10 27 CD FF :FF
CA80 18 C9 CD B1 28 CD 28 14 :90
CA88 3A 81 2F FE 7D 20 F6 C9 :44
CA90 CD E7 18 CD 5C 1B FE 7B :89
CA98 C2 B0 2D 3E B7 CD F5 26 :7C
CAA0 3E CA 21 00 00 CD 10 27 :2D
CAA8 2A 57 2F 2B 2B 22 C7 2F :1E
CAB0 CD 2A 15 2A C5 2F 3E 18 :80
CAB8 CD CC 18 3E C3 DC 10 27 :C5
CAC0 CD FF 18 C3 B1 28 3D C2 :7F
CAC8 B0 2D 21 61 2F CD 57 0C :BE
CAD0 B7 C4 16 2C 21 61 2F 7E :EC
CAD8 E6 2F FE 02 C2 B0 2D 01 :B5
CAE0 0F 00 CD 95 28 21 61 2F :4A
CAE8 CB EE CD D0 0B 01 06 00 :68
CAF0 21 C9 2F CD 95 28 2A 6E :3B
CAF8 2F 22 C9 2F CD B1 28 FE :ED
CB00 F0 C2 B0 2D CD B1 28 CD :02
CB08 5C 1B FE 99 C2 B0 2D 3E :EB
CB10 32 2A C9 2F CD 10 27 CD :25
CB18 B1 28 CD 5C 1B 2A 5B 2F :D1
CB20 E5 22 CB 2F CD 7F 16 3E :A1
CB28 32 2B CD 10 27 21 00 00 :82
CB30 22 CD 2F 3A 81 2F FE 7B :81
CB38 28 1F FE 9A C2 B0 2D CD :4B
CB40 B1 28 CD 5C 1B FE 7B C2 :58
CB48 B0 2D 2A 5B 2F 22 CD 2F :AF
CB50 CD 7F 16 3E 32 2B CD 10 :DA
CB58 27 CD E7 18 2A CB 2F CD :E4
CB60 DF 25 2A C9 2F CD DF 25 :F7
CB68 21 90 16 CD 3C 26 2A 57 :77
CB70 2F 2B 2B 22 C7 2F CD 2A :94
CB78 15 2A CD 2F 7C B5 20 19 :A5
CB80 2A C9 2F 3E 21 CD 10 27 :85
CB88 3E 34 CD F5 26 2A C5 2F :78
CB90 3E 20 CD CC 18 3E C2 18 :27
CB98 19 CD DF 25 2A C9 2F CD :D9
CBA0 DF 25 21 9B 16 CD 3C 26 :05
CBA8 2A C5 2F 3E 30 CD CC 18 :3D
CBB0 3E D2 DC 10 27 CD FF 18 :07
CBB8 E1 22 5B 2F 01 06 00 21 :B5
CBC0 C9 2F CD A5 28 01 0F 00 :A2
CBC8 21 61 2F CD A5 28 21 61 :CD
CBD0 2F CD D0 0B C3 B1 28 23 :96
CBD8 22 5B 2F EB 2A 59 2F B7 :00
```

```
CBE0 ED 52 EB D0 22 59 2F C9 :6D
CBE8 0A 3A B9 B9 21 BB BB 96 :E3
CBF0 DA 00 00 08 3A B9 B9 21 :AF
CBF8 BB BB 86 77 FE 23 CA 2E :8C
CC00 17 3D C2 B0 2D 21 61 2F :A4
CC08 CD 57 0C B7 C4 16 2C 21 :0E
CC10 61 2F 7E E6 2F FE 02 C2 :E5
CC18 B0 2D 01 0F 00 CD 95 28 :77
CC20 21 61 2F CB EE CD D0 0B :12
CC28 2A C9 2F E5 2A 6E 2F 22 :F0
CC30 C9 2F CD B1 28 FE 2C C2 :8A
CC38 B0 2D CD B1 28 CD 5C 1B :C7
CC40 FE 7B C2 B0 2D 3E 32 2A :B2
CC48 C9 2F CD 10 27 CD E7 18 :C8
CC50 CD 2A 15 3E 21 2A C9 2F :8D
CC58 CD 10 27 3E 35 CD F5 26 :5F
CC60 2A C5 2F 3E 20 CD CC 18 :2D
CC68 3E C2 DC 10 27 CD FF 18 :F7
CC70 E1 22 C9 2F 01 0F 00 21 :2C
CC78 61 2F CD A5 28 21 61 2F :DB
CC80 CD D0 0B C3 B1 28 21 84 :E9
CC88 2F 7E B7 C2 B0 2D 2F 77 :A9
CC90 CD B1 28 FE 2C C2 B0 2D :6F
CC98 CD B1 28 CD 97 1B 47 3A :A6
CCA0 86 2F 3D C2 DF 2D 78 FE :36
CCA8 7B C2 B0 2D FD 7E FD B7 :49
CCB0 28 11 21 8E 1B 3D 28 08 :70
CCB8 21 96 17 CD 31 26 18 08 :12
CCC0 CD 31 26 3E 47 CD F5 26 :91
CCC8 CD E7 18 CD 2A 15 2A C5 :C7
CCD0 2F 3E 10 CD CC 18 30 0C :6A
CCD8 E5 3E 05 CD F5 26 E1 3E :2F
CCE0 C2 CD 10 27 AF 32 84 2F :5A
CCE8 CD FF 18 C3 B1 28 02 06 :88
CCF0 B8 2A D1 2F E5 2A CF 2F :EF
CCF8 E5 21 00 00 22 D1 2F 22 :4A
CD00 CF 2F CD 5C 1B FE 94 C2 :96
CD08 B0 2D 3E B7 CD F5 26 3E :F8
CD10 CA 21 00 00 CD 10 27 2A :19
CD18 57 2F 2B 2B 22 CF 2F CD :C9
CD20 B1 28 CD 28 14 3A 81 2F :CC
CD28 FE BD 28 2A FE 95 28 08 :D0
CD30 2A CF 2F CD 59 27 18 0D :9A
CD38 CD B1 28 FE 93 28 17 CD :43
CD40 06 18 CD 28 14 2A D1 2F :51
CD48 7C B5 C4 59 27 E1 22 CF :47
CD50 2F E1 22 D1 2F C9 CD 06 :CE
CD58 18 CD B1 28 18 A4 2A D1 :75
CD60 2F 3E C3 CD 10 27 2A CF :2D
CD68 2F CD 59 27 2A 57 2F 2B :57
CD70 2B 22 D1 2F C9 FE 3B C2 :11
CD78 B0 2D 3E C3 2A C7 2F CD :CB
CD80 10 27 2A 57 2F 2B 2B 22 :5F
CD88 C7 2F C3 B1 28 FE 99 C2 :EB
CD90 B0 2D CD B1 28 3D C2 B0 :32
CD98 2D 21 61 2F CD 57 0C B7 :C5
CDA0 28 0D 3E 98 32 61 2F 21 :EE
CDA8 00 00 22 6E 2F 18 1E 3A :2F
CDB0 61 2F 47 E6 0F FE 08 C2 :94
CDB8 B0 2D 2A 6E 2F 78 87 38 :DB
CDC0 0C 3E 18 CD CC 18 3E C3 :14
CDC8 DC 10 27 18 13 3E C3 CD :0C
CDD0 10 27 2A 57 2F 2B 2B 22 :5F
CDD8 6E 2F 21 61 2F CD D0 0B :F6
CDE0 CD B1 28 FE 3B C2 B0 2D :7E
CDE8 C3 B1 28 FE 3B C2 B0 2D :74
CDF0 3A D7 2F B7 28 12 3E C3 :32
CDF8 2A D8 2F CD 10 27 2A 57 :B6
CE00 2F 2B 2B 22 D8 2F 18 05 :CB
CE08 3E C9 CD F5 26 C3 B1 28 :8B
CE10 FE 3B C2 B0 2D 3E C3 2A :03
CE18 51 2F 11 03 00 19 CD 10 :8A
CE20 27 C3 B1 28 E5 F5 ED 5B :E5
CE28 57 2F 13 13 B7 ED 52 D1 :73
CE30 5D 7D 87 9F BC E1 20 05 :C2
CE38 CD 06 27 B7 C9 37 C9 DD :57
CE40 E1 2A C5 2F E5 2A C7 2F :04
CE48 E5 21 00 00 22 C7 2F 2A :48
CE50 57 2F 22 C5 2F DD E9 DD :3F
CE58 E1 2A C7 2F 7C B5 C4 59 :4F
CE60 27 E1 22 C7 2F E1 22 C5 :E8
CE68 2F DD E9 3A 81 2F D6 A1 :56
CE70 DA 51 1B FE 05 D2 51 1B :87
CE78 87 5F 16 00 21 2F 19 19 :7E
CE80 5E 23 56 D5 C3 B1 28 39 :81
CE88 19 E6 19 8F 1A 3F 1B 42 :5D
CE90 1B D6 BF 28 05 FE 01 C2 :9E
CE98 C4 2D F5 CD B1 28 FE F0 :7A
CEA0 C2 C4 2D CD B1 28 D6 BF :EE
CEA8 28 0B FE 01 28 07 CD A3 :D1
CEB0 27 3E 02 18 05 F5 CD B1 :F7
CEB8 28 F1 D1 5F 3A 81 2F FE :31
CEC0 3B 28 43 FE 2B 28 05 D6 :D2
CEC8 2D C2 C4 2D F5 D5 E5 CD :5C
CED0 B1 28 CD 51 1B E1 D1 D5 :99
CED8 CD B4 19 D1 15 16 DD 20 :93
CEE0 02 16 FD 1E 19 F1 D5 B7 :C9
CEE8 28 0A 3E 5F 21 16 00 CD :D3
CEF0 10 27 18 0E 11 5F 2F CD :C9
CEF8 06 27 3E 16 21 FF 13 CD :81
CF00 10 27 D1 C3 06 27 CD B4 :79
CF08 19 C3 B1 28 7B BA C8 B7 :69
CF10 20 0C 11 E5 DD CD 06 27 :F9
CF18 11 E1 FD C3 06 27 3D 20 :3C
CF20 0C 11 E5 FD CD 06 27 11 :0A
CF28 E1 DD C3 06 27 15 3E DD :DE
CF30 20 02 3E FD E5 CD F5 26 :2A
CF38 E1 3E 21 C3 10 27 16 DD :2D
CF40 D6 BF 28 06 16 FD 3D C2 :D5
CF48 C4 2D D5 CD B1 28 FE F0 :5A
CF50 C2 C4 2D CD B1 28 3D C2 :58
CF58 C4 2D 21 61 2F CD 57 0C :D2
```

```
CF60 B7 C4 16 2C 2A 6E 2F 3A :BE
CF68 61 2F E6 0F D6 02 28 04 :89
CF70 3D C2 C4 2D E5 CD B1 28 :7B
CF78 FE 5B 20 44 CD B1 28 CD :30
CF80 97 1B 3A 86 2F 3D C2 DF :7F
CF88 2D 3A 81 2F FE 5D C2 C4 :F8
CF90 2D CD B1 28 FD 7E FD FE :49
CF98 02 28 1B B7 28 06 21 7F :CA
CFA0 1A CD 31 26 E1 CD DF 25 :F0
CFA8 21 83 1A CD 31 26 D1 1E :D1
CFB0 E1 CD 06 27 18 16 E1 FD :E7
CFB8 7E FE 85 6F 30 01 24 FE :C3
CFC0 E1 E3 EB 1E 2A CD 06 27 :F1
CFC8 E1 CD 08 27 3A 81 2F FE :C5
CFD0 3B CA B1 28 C3 C4 2D 03 :95
CFD8 3A B9 B9 0B 5F 16 00 21 :4D
CFE0 B9 B9 19 5E 23 56 D5 3D :74
CFE8 C2 C4 2D 21 61 2F CD 57 :88
CFF0 0C B7 C4 16 2C 2A 6E 2F :90
CFF8 3A 61 2F E6 0F FE 02 C2 :81
D000 C4 2D E5 CD B1 28 FE 5B :D5
D008 20 5C CD B1 28 CD 97 1B :A1
D010 3A 86 2F 3D C2 DF 2D 3A :34
D018 81 2F FE 5D C2 C4 2D FD :BB
D020 7E FD FE 02 28 33 B7 28 :B5
D028 06 21 7F 1A CD 31 26 E1 :C5
D030 CD DF 25 21 32 1B CD 31 :3D
D038 26 CD B1 28 FE F0 C2 C4 :40
D040 2D CD B1 28 16 DD FE BF :83
D048 28 02 16 FD 1E E5 CD 06 :13
D050 27 21 3A 1B CD 31 26 18 :D9
D058 26 E1 FD 7E FE 85 6F 30 :A4
D060 01 24 E5 CD B1 28 FE F0 :9E
D068 C2 C4 2D CD B1 28 16 DD :4C
D070 FE BF 28 02 16 FD 1E 22 :3A
D078 CD 06 27 E1 CD 08 27 CD :A4
D080 B1 28 FE 3B CA B1 28 C3 :78
D088 C4 2D 07 5F 16 00 21 B9 :47
D090 B9 19 04 D1 73 23 72 1E :CD
D098 23 21 1E 2B 16 DD FE BF :3D
D0A0 28 02 16 FD CD 06 27 18 :4F
D0A8 D6 CD 5F 1B FE 3B CA B1 :D1
D0B0 28 C3 DF 2D CD D3 25 CD :89
D0B8 71 1B 3A 86 2F 3D C2 DF :59
D0C0 2D 3A 81 2F C9 AF 32 85 :46
D0C8 2F CD 97 1B FD 7E FD B7 :DD
D0D0 C8 21 8E 1B 3D 28 0C 21 :24
D0D8 95 1B FD 7E FE B7 28 03 :0B
D0E0 21 92 1B C3 31 26 FD 3A :1F
D0E8 B9 B9 FE 3E B8 FF AF CD :E1
D0F0 49 1C 3A 81 2F FE A6 28 :1B
D0F8 21 FE A7 28 07 FE 21 28 :3C
D100 03 FE 7C C0 CD 46 1C 21 :8D
D108 D8 1B CD 18 26 30 E3 FD :0E
D110 7E FE FD B6 01 FD 77 FE :A2
D118 18 D8 CD 46 1C 21 0F 1C :6B
D120 CD 18 26 30 CD FD 7E FE :81
D128 FD AE 01 FD 77 FE 18 C2 :F8
D130 E8 1B EB 1B F0 1B F3 1B :22
D138 F8 1B 00 1C 06 1C 09 1C :76
D140 02 D1 B2 04 21 BB BB B6 :D6
D148 02 F6 BA 04 21 B9 B9 B6 :FF
D150 F9 3A B9 B9 21 BB BB B6 :F2
D158 FB 3A B9 B9 F6 BA 02 F6 :4F
D160 B8 FB 3A BB BB F6 B8 1F :30
D168 1C 22 1C 27 1C 2A 1C 2F :12
D170 1C 37 1C 3D 1C 40 1C 02 :26
D178 D1 AA 04 21 BB BB AE 02 :C6
D180 EE BA 04 21 B9 B9 AE F9 :E6
D188 3A B9 B9 21 BB BB AE FB :EC
D190 3A B9 B9 EE BA 02 EE B8 :FC
D198 FB 3A BB BB EE B8 CD B1 :CF
D1A0 28 CD A6 1C 3A 81 2F FE :9F
D1A8 A8 28 03 FE 26 C0 CD A3 :27
D1B0 1C 21 6C 1C CD 18 26 30 :00
D1B8 EB FD 7E FE FD A6 01 FD :05
D1C0 77 FE 18 E0 7C 1C 7F 1C :A0
D1C8 84 1C 87 1C 8C 1C 94 1C :9B
D1D0 9A 1C 9D 1C 02 D1 A2 04 :E8
D1D8 21 BB BB A6 02 E6 BA 04 :E3
D1E0 21 B9 B9 A6 F9 3A B9 B9 :DE
D1E8 21 BB BB A6 FB 3A B9 B9 :E4
D1F0 E6 BA 02 E6 B8 FB 3A BB :30
D1F8 BB E6 B8 CD B1 28 3A 81 :BA
D200 2F FE A9 28 08 FE 5E 28 :8A
D208 04 FE 7E 20 27 CD B1 28 :6D
D210 CD DC 1C FD 7E FD B7 28 :1C
D218 11 3D 28 08 FD 7E FE 2F :26
D220 FD 77 FE C9 21 D7 1C C3 :12
D228 31 26 3E 2F C3 F5 26 FC :9E
D230 3A B9 B9 2F CD 5C 1D 3A :5B
D238 81 2F FE 3D 28 57 FE F3 :5B
D240 28 42 FE 3E 28 2D FE F4 :ED
D248 28 1A FE 3C 28 0E FE F2 :A2
D250 C0 CD 59 1D CD 05 26 CD :C8
D258 9F 1D 18 10 CD 59 1D CD :F4
D260 9F 1D 18 1A CD 59 1D CD :FE
D268 9F 1D 06 02 3F 9F 21 13 :D6
D270 1D 18 32 CD 59 1D CD 05 :7C
D278 26 CD 9F 1D 06 01 9F 21 :76
D280 25 1D 18 21 CD 59 1D CD :8B
D288 9F 1D 06 04 28 02 3E FF :2D
D290 21 33 1D 18 10 CD 59 1D :DC
D298 CD 9F 1D 06 05 28 02 3E :FC
D2A0 FF 2F 21 44 1D 47 FD 7E :72
D2A8 FD B7 CA 31 26 FD 70 FE :40
D2B0 C9 CD B1 28 CD B9 1E 3A :4D
D2B8 81 2F FE 2B 28 4A FE 2D :76
D2C0 28 2D FE AA 28 16 FE AB :E4
D2C8 C0 CD B6 1E 21 C6 1D CD :32
D2D0 18 26 30 E3 21 02 1E CD :5F
D2D8 3C 26 18 DB CD B6 1E 21 :17
```

```
D2E0 07 1E CD 18 26 30 D0 21 :51
D2E8 3E 1E CD 3C 26 18 C8 CD :38
D2F0 B6 1E CD 9F 1D 18 C0 21 :56
D2F8 43 1E CD 18 26 D0 FD 7E :B7
D300 FE FD 96 01 FD 77 FE C9 :CD
D308 CD B6 1E 21 7F 1E CD 18 :44
D310 26 30 A4 FD 7E FE FD 86 :F6
D318 01 FD 77 FE 18 99 D6 1D :17
D320 DA 1D DF 1D E2 1D E8 1D :F7
D328 F0 1D F6 1D FB 1D 03 57 :92
D330 F1 9A 04 21 BB BB 9E 02 :C6
D338 DE BA 05 57 3A B9 B9 9A :3A
D340 F9 3A B9 B9 21 BB BB 9E :DA
D348 FB 3A B9 B9 DE BA 04 57 :9A
D350 3E B8 9A FA 3E B8 21 BB :5C
D358 BB 9E FC 3E B8 DE BA 17 :FA
D360 1E 1A 1E 1F 1E 22 1E 27 :FA
D368 1E 2F 1E 35 1E 38 1E 02 :16
D370 D1 8A 04 21 BB BB 8E 02 :86
D378 CE BA 04 21 B9 B9 8E F9 :A6
D380 3A B9 B9 21 BB BB 8E FB :CC
D388 3A B9 B9 CE BA 02 CE BA :BE
D390 FB 3A BB BB CE B8 FC 3E :6B
D398 B8 CE BA 53 1E 57 1E 5C :82
D3A0 1E 5F 1E 65 1E 6D 1E 73 :1C
D3A8 1E 78 1E 03 57 F1 92 04 :95
D3B0 21 BB BB 96 02 D6 BA 05 :C4
D3B8 57 3A B9 B9 92 F9 3A B9 :81
D3C0 B9 21 BB BB 96 FB 3A B9 :D4
D3C8 B9 D6 BA 04 57 3E B8 92 :2C
D3D0 FA 3E B8 21 BB BB 96 8F :AC
D3D8 1E 92 1E 97 1E 9A 1E 9F :DA
D3E0 1E A7 1E AD 1E B0 1E 02 :7E
D3E8 D1 82 04 21 BB BB 86 02 :76
D3F0 C6 BA 04 21 B9 B9 86 F9 :96
D3F8 3A B9 B9 21 BB BB 86 FB :C4
D400 3A B9 B9 C6 BA 02 C6 B8 :AC
D408 FB 3A BB BB C6 B8 CD B1 :A7
D410 28 CD D7 20 3A 81 2F FE :D4
D418 2A 28 73 FE 2F 28 56 FE :6E
D420 25 28 39 FE F1 28 1C FE :B7
D428 F5 C0 CD D4 20 21 51 1F :07
D430 CD 18 26 30 DF FD 7E FE :93
D438 FD 56 01 CD 1C 2E FD 77 :DF
D440 FE 18 D1 CD D4 20 21 9F :68
D448 1F CD 18 26 30 C6 FD 7E :9B
D450 FE FD 56 01 CD 16 2E FD :60
D458 77 FE 18 B8 CD D4 20 21 :27
D460 ED 1F CD 18 26 30 AD FD :F1
D468 7E FE FD 56 01 CD 10 2E :DB
D470 FD 77 FE 18 9F CD D4 20 :EA
D478 21 3B 20 CD 18 26 30 94 :4B
D480 FD 7E FE FD 56 01 CD 0A :A4
D488 2E FD 77 FE 18 86 CD D4 :DF
D490 20 21 89 20 CD 18 26 D2 :C7
D498 BC 1E FD 7E FE FD 56 01 :A7
D4A0 CD 04 2E FD 77 FE C3 BC :F0
D4A8 1E 61 1F 67 1F 6E 1F 74 :25
D4B0 1F 7C 1F 86 1F 8F 1F 96 :A3
D4B8 1F 05 57 F1 CD 21 00 06 :60
D4C0 21 BB BB CD 1E 00 05 16 :9D
D4C8 BA CD 21 00 07 57 3A B9 :F9
D4D0 B9 CD 21 00 F7 3A B9 B9 :4A
D4D8 21 BB BB CD 1E 00 F8 3A :B4
D4E0 B9 B9 16 BA CD 21 00 06 :36
D4E8 57 3E B8 CD 21 00 F8 3E :71
D4F0 B8 21 BB BB CD 1E 00 AF :E9
D4F8 1F B5 1F BC 1F C2 1F CA :79
D500 1F D4 1F DD 1F E4 1F 05 :16
D508 57 F1 CD 1B 00 06 21 BB :12
D510 BB CD 18 00 05 16 BA CD :42
D518 1B 00 07 57 3A B9 B9 CD :F2
D520 1B 00 F7 3A B9 B9 21 BB :9A
D528 BB CD 18 00 F8 3A B9 B9 :44
D530 16 BA CD 1B 00 06 57 3E :53
D538 B8 CD 1B 00 F8 3E B8 21 :AF
D540 BB BB CD 18 00 FD 1F 03 :7A
D548 20 0A 20 10 20 18 20 22 :D4
D550 20 2B 20 32 20 05 57 F1 :0A
D558 CD 15 00 06 21 BB BB CD :4C
D560 12 00 05 16 BA CD 15 00 :C9
D568 07 57 3A B9 B9 CD 15 00 :EC
D570 F7 3A B9 B9 21 BB BB CD :07
D578 12 00 F8 3A B9 B9 16 BA :86
D580 CD 15 00 06 57 3E B8 CD :02
D588 15 00 F8 3E B8 21 BB BB :9A
D590 CD 12 00 4B 20 51 20 58 :13
D598 20 5E 20 66 20 70 20 79 :2D
D5A0 20 80 20 05 57 F1 CD 0F :E9
D5A8 00 06 21 BB BB CD 0C 00 :76
D5B0 05 16 BA CD 0F 00 07 57 :0F
D5B8 3A B9 B9 CD 0F 00 F7 3A :B9
D5C0 B9 B9 21 BB BB CD 0C 00 :E2
D5C8 F8 3A B9 B9 16 BA CD 0F :50
D5D0 00 06 57 3E B8 CD 0F 00 :2F
D5D8 F8 3E B8 21 BB BB CD 0C :5E
D5E0 00 99 20 9E 20 A5 20 AB :E7
D5E8 20 B2 20 BC 20 C5 20 CB :7E
D5F0 20 04 D1 CD 09 00 06 21 :F2
D5F8 BB BB CD 06 00 05 16 BA :1E
D600 CD 09 00 06 21 B9 B9 CD :3C
D608 06 00 F7 3A B9 B9 21 BB :85
D610 BB CD 06 00 F8 3A B9 B9 :32
D618 16 BA CD 09 00 05 16 B8 :79
D620 CD 09 00 F8 3A BB BB 16 :94
D628 B8 CD 09 00 CD B1 28 3A :6E
D630 81 2F FE 2B 28 2B FE 2D :57
D638 20 2A CD 09 21 FD 7E FD :B9
D640 B7 28 18 3D 28 09 FD 7E :E0
D648 FE ED 44 FD 77 FE C9 21 :8B
D650 FD 20 C3 31 26 FB 3A B9 :25
D658 B9 ED 44 11 44 ED C3 06 :F5
```

```
D660 27 CD B1 28 FE 28 20 26 :39
D668 CD B1 28 CD 97 1B FE 2C :4F
D670 20 14 CD 74 1B CD F8 25 :7A
D678 CD B1 28 CD 6D 1B 3A 81 :B6
D680 2F 18 EB 3A 81 2F FE 29 :43
D688 C2 DF 2D C3 B1 28 FE 3E :A7
D690 20 53 CD B1 28 FE 28 C2 :01
D698 DF 2D CD B1 28 CD 71 1B :0B
D6A0 3A 81 2F FE 3B C2 DF 2D :F1
D6A8 3E B7 CD F5 26 23 E5 3E :23
D6B0 CA 21 00 00 CD 10 27 CD :BC
D6B8 B1 28 CD 6D 1B CD F8 25 :18
D6C0 3A 81 2F FE 2C C2 DF 2D :E2
D6C8 3E C3 21 00 00 CD 10 27 :26
D6D0 2B 2B E3 CD 59 27 CD B1 :04
D6D8 28 CD 6D 1B CD F8 25 E1 :48
D6E0 CD 59 27 18 9E FE B9 28 :E2
D6E8 05 FE 40 C2 95 22 CD B1 :3A
D6F0 28 FE 5B C2 DF 2D CD B1 :CD
D6F8 28 FE BF 28 4B FE C0 28 :3E
D700 4A CD 71 1B CD F8 25 3A :C7
D708 81 2F FE 2C C2 DF 2D CD :75
D710 B1 28 3E F5 CD F5 26 CD :C1
D718 6D 1B 3A 81 2F FE 5D C2 :8F
D720 DF 2D 11 6F E1 CD 06 27 :67
D728 CD B1 28 FE F0 20 14 CD :95
D730 F8 25 3E E5 CD F5 26 CD :F5
D738 B1 28 CD 6D 1B 11 77 E1 :97
D740 C3 06 27 3E 7E C3 F5 26 :8A
D748 3E DD 21 3E FD F5 CD B1 :EA
D750 28 FE 2B 28 16 FE 2D 28 :E2
D758 0F 2E 00 FE 2C 28 1E 26 :D3
D760 00 FE 5D 28 4C C3 DF 2D :9E
D768 2E 2B 11 2E 23 E5 CD B1 :1E
D770 28 E1 26 00 FE 5D 28 39 :EB
D778 FE 2C C2 DF 2D E5 CD B1 :5B
D780 28 FE 2B 28 05 FE 2D 20 :C9
D788 08 FE AF F5 CD B1 28 F1 :41
D790 0E AF F5 CD 8F 27 F1 28 :4E
D798 04 AF 95 6F 9F CB 7D 28 :C6
D7A0 01 2F B7 C4 3B 2D D1 55 :39
D7A8 EB 3A 81 2F FE 5D C2 DF :D1
D7B0 2D E5 CD B1 28 FE F0 20 :C6
D7B8 19 CD B1 28 CD 71 1B E1 :F9
D7C0 D1 7A D5 E5 2E 77 CD 10 :87
D7C8 27 E1 D1 7D B7 C8 5D C3 :F5
D7D0 06 27 CD E6 26 E1 D1 7A :32
D7D8 D5 E5 2E 7E CD 10 27 E1 :4B
D7E0 D1 7D B7 28 05 5D CD 06 :62
D7E8 27 AF C3 DF 25 FE BA 28 :7D
D7F0 04 FE F6 20 3F CD B1 28 :FD
D7F8 FE 5B C2 DF 2D CD B1 28 :CD
D800 CD 71 1B 3A 81 2F FE 5D :9E
D808 C2 DF 2D CD B1 28 FE F0 :62
D810 20 1A CD F8 25 3E F5 CD :24
D818 F5 26 CD B1 28 CD 6D 1B :16
D820 11 4A D1 CD 06 27 11 79 :B0
D828 ED C3 06 27 3E 4F 21 ED :78
D830 78 C3 10 27 FE AC 28 07 :4B
D838 FE AD 20 48 3E 01 FE AF :FF
D840 F5 CD B1 28 FE 28 C2 DF :62
D848 2D CD B1 28 3D C2 DF 2D :DE
D850 21 61 2F CD 57 0C B7 C4 :5C
D858 16 2C 3A 61 2F E6 0F FE :FF
D860 02 C2 DF 2D 2A 6E 2F 3E :D5
D868 21 CD 10 27 F1 C6 34 CD :DD
D870 F5 26 AF CD DF 25 21 2A :E6
D878 23 CD 31 26 CD B1 28 C3 :B0
D880 2B 21 FF 7E FE AE DA C0 :0F
D888 23 FE B4 30 2E D6 AE F5 :AC
D890 CD B1 28 FE 28 C2 DF 2D :9A
D898 CD B1 28 CD 13 21 CD 74 :E8
D8A0 1B F1 5F 16 00 21 5D 23 :22
D8A8 19 7E FE CB C2 F5 26 57 :94
D8B0 1E 2F C3 06 27 17 07 1F :7A
D8B8 0F CB 27 FE B9 30 59 D6 :17
D8C0 B4 F5 CD B1 28 FE 28 C2 :37
D8C8 DF 2D CD B1 28 FE 29 28 :01
D8D0 08 CD 13 21 CD 74 1B 18 :7D
D8D8 0A CD B1 28 CD E6 26 AF :38
D8E0 CD DF 25 F1 20 04 3E 9F :C3
D8E8 18 13 F5 11 00 3E CD 06 :42
D8F0 27 F1 3D 20 0B 11 01 20 :B2
D8F8 CD 06 27 3E 2F C3 F5 26 :45
D900 21 00 00 3D 3E F2 28 02 :B8
D908 3E E2 CD 10 27 3E 2F CD :5E
D910 F5 26 2B 2B 2B C3 59 27 :DF
D918 2A 5D 2F B7 CA 97 24 3D :2F
D920 C2 92 24 21 61 2F CD 57 :4D
D928 0C B7 20 19 2A 6E 2F 3A :FD
D930 61 2F E6 0F FE 01 CA 97 :E5
D938 24 FE 08 CA DF 2D FE 09 :07
D940 CA CF 24 18 13 2A 4B 2F :8C
D948 CD 60 2B FE 28 CA A6 24 :12
D950 CD 16 2C 3E 02 21 00 00 :70
D958 F5 E5 CD B1 28 FE 5B 20 :F9
D960 62 CD B1 28 CD 97 1B 3A :C1
D968 81 2F FE 5D C2 DF 2D FD :D6
D970 7E FD FE 02 28 3D B7 28 :BF
D978 06 21 8E 1B CD 31 26 E1 :D5
D980 CD DF 25 21 8A 24 CD 31 :9E
D988 26 CD F8 25 CD B1 28 FE :B4
D990 F0 20 1A F1 FE 03 CA DF :C5
D998 2D CD F8 25 3E E5 CD F5 :FC
D9A0 26 CD B1 28 CD 6D 1B 11 :32
D9A8 77 E1 C3 06 27 F1 3E 7E :F5
D9B0 C3 F5 26 E1 FD 7E FE 85 :BD
D9B8 6F 30 01 24 E5 CD F8 25 :93
D9C0 CD B1 28 FE F0 20 14 E1 :A9
D9C8 F1 FE 03 CA DF 2D E5 CD :7A
D9D0 B1 28 CD 71 1B E1 3E 32 :83
D9D8 C3 10 27 E1 F1 3E 01 C3 :CE
```

```
D9E0 DF 25 07 5F 16 00 21 B9 :5A
D9E8 B9 19 CD 8F 27 18 0A 7C :F3
D9F0 B7 C4 4D 2D E5 CD B1 28 :80
D9F8 E1 3E 02 C3 DF 25 21 00 :09
DA00 00 22 C3 2F 01 0C 00 21 :42
DA08 62 2F CD 95 28 CD D8 24 :E4
DA10 01 0C 00 21 62 2F CD A5 :31
DA18 28 2A C3 2F 22 6E 2F 21 :24
DA20 61 2F 36 89 C3 D0 0B 3A :27
DA28 61 2F 07 38 D4 22 C3 2F :B7
DA30 CD B1 28 FE 28 C2 DF 2D :9A
DA38 CD E6 26 3A 84 2F B7 F5 :72
DA40 3E C5 C4 F5 26 2A C3 2F :FE
DA48 E5 CD B1 28 FE 3B 28 4D :39
DA50 FE 29 CA AA 25 11 01 36 :08
DA58 D5 18 04 D5 CD B1 28 CD :39
DA60 6D 1B CD F8 25 3E F5 CD :72
DA68 F5 26 D1 3A 81 2F FE 3B :0F
DA70 28 16 FE 29 28 12 FE 2C :C9
DA78 C2 DF 2D 1C 7B FE 21 38 :BC
DA80 DA CC 01 2D 1E 21 18 D3 :FE
DA88 F5 D5 3E 21 2A 53 2F CD :A2
DA90 10 27 D1 CD 06 27 F1 FE :F1
DA98 29 28 74 18 0D 3E AF CD :A4
DAA0 F5 26 3E 32 2A 53 2F CD :04
DAA8 10 27 CD B1 28 FE 2C 28 :2F
DAB0 28 FE BF 20 10 11 E5 DD :E8
DAB8 CD 06 27 3E E1 CD F5 26 :01
DAC0 CD B1 28 18 08 CD 69 27 :23
DAC8 3E 21 CD 10 27 3A 81 2F :4D
DAD0 FE 29 28 3B FE 2C C2 DF :55
DAD8 2D CD B1 28 FE C0 20 10 :C1
DAE0 11 E5 FD CD 06 27 3E D1 :FC
DAE8 CD F5 26 CD B1 28 18 08 :AE
DAF0 CD 69 27 3E 11 CD 10 27 :B0
DAF8 3A 81 2F FE 29 C2 DF 2D :DF
DB00 18 0D 3E AF CD F5 26 3E :38
DB08 32 2A 53 2F CD 10 27 D1 :B3
DB10 2A 57 2F 23 22 C3 2F EB :D2
DB18 3E CD CD 10 27 F1 3E C1 :FF
DB20 C4 F5 26 3E 00 CD DF 25 :EE
DB28 C3 B1 28 FD 21 87 2F AF :1F
DB30 32 86 2F 32 85 2F C9 FD :93
DB38 77 00 FD 75 01 FD 74 02 :5D
DB40 11 03 00 FD 19 21 86 2F :00
DB48 34 7E FE 14 D2 DF 2D C9 :6B
DB50 FD 2B FD 2B FD 2B E5 21 :7E
DB58 86 2F 35 E1 C9 FD E5 D1 :47
DB60 21 FD FF 19 06 03 2B 1B :85
DB68 1A 4E EB 71 12 10 F7 C9 :A6
DB70 FD 7E FA 5F 87 83 FD 86 :61
DB78 FD FE 08 CA CD 26 87 5F :A6
DB80 16 00 19 5E 23 56 EB 18 :09
DB88 0B 11 03 00 FD 19 EB 21 :41
DB90 86 2F 34 EB 7E 47 B7 FC :4C
DB98 D2 26 C5 E5 CD BB 0B E1 :16
DBA0 C1 23 C5 7E FE B8 20 05 :02
DBA8 FD 7E FB 18 62 FE B9 20 :C7
DBB0 0C FD 7E FB E5 CD BB 0B :FA
DBB8 FD 7E FC 18 17 FE BA 20 :7E
DBC0 05 FD 7E FE 18 49 FE BB :98
DBC8 20 1B FD 7E FE E5 CD BB :21
DBD0 0B FD 7E FF CD BB 0B 2A :42
DBD8 57 2F 23 23 22 57 2F E1 :55
DBE0 23 C1 05 18 30 FE CD 20 :1C
DBE8 26 E5 CD BB 0B E1 23 5E :00
DBF0 23 56 E5 2A 51 2F 19 E5 :06
DBF8 7D CD BB 0B F1 CD BB 0B :94
DC00 2A 57 2F 23 23 23 22 57 :92
DC08 2F E1 C1 05 05 18 06 E5 :DE
DC10 CD FB 26 E1 C1 10 8A FD :27
DC18 36 FA 00 3E FF 32 85 2F :53
DC20 CD F8 25 B7 C9 CD F8 25 :54
DC28 37 C9 ED 44 47 3A 85 2F :66
DC30 B7 78 C8 3C C5 E5 CD FB :A5
DC38 26 E1 C1 3E F5 C9 3A 85 :83
DC40 2F B7 3E F5 C2 F5 26 3E :34
DC48 FF 32 85 2F C9 F5 AF CD :1F
DC50 BB 0B F1 CD BB 0B 2A 57 :CB
DC58 2F 23 22 57 2F C9 63 6A :90
DC60 E5 3E 02 CD BB 0B 18 0B :DB
DC68 E5 F5 3E 03 CD BB 0B F1 :9F
DC70 CD FB 26 E1 7D E5 CD BB :B9
DC78 0B F1 CD BB 0B 2A 57 2F :3F
DC80 23 23 22 57 2F C9 21 62 :3A
DC88 2F 11 71 2F 06 0C 7E 12 :82
DC90 B7 C8 23 13 10 F8 78 12 :47
DC98 C9 CD BB 0B 21 71 2F 7E :9B
DCA0 B7 CA BB 0B E5 CD BB 0B :BF
DCA8 E1 23 18 F3 E5 3E 80 18 :CA
DCB0 03 E5 3E 81 CD BB 0B E1 :1B
DCB8 7D E5 CD BB 0B F1 C3 BB :64
DCC0 0B CD A3 27 3A 81 2F FE :8A
DCC8 2B 28 11 FE 2D C0 E5 CD :01
DCD0 B1 28 CD A3 27 EB E1 B7 :F3
DCD8 ED 52 18 E8 E5 CD B1 28 :CA
DCE0 CD A3 27 D1 19 18 DD 21 :97
DCE8 83 2F 7E F5 36 00 CD A3 :CB
DCF0 27 F1 32 83 2F 7C B7 C2 :F1
DCF8 4D 2D C9 21 83 2F 46 3A :96
DD00 81 2F FE BB 28 07 FE BC :52
DD08 28 0F C3 D1 27 C5 36 00 :ED
DD10 CD B1 28 CD D1 27 6C 18 :EF
DD18 09 C5 36 00 CD B1 28 CD :77
DD20 D1 27 F1 32 83 2F 26 00 :F3
DD28 C9 3A 81 2F B7 20 05 2A :B9
DD30 5D 2F 18 43 3D 20 15 21 :7A
DD38 61 2F CD 57 0C B7 C4 28 :63
DD40 2C 3A 61 2F E6 0F 3D C2 :EA
DD48 80 2D 18 28 FE 2D C2 80 :5A
DD50 2D CD B1 28 3D C2 94 2D :93
DD58 21 61 2F CD 57 0C B7 20 :B8
```

```
DD60 49 3A 61 2F E6 0F 3D CA :0F
DD68 94 2D 3D 28 07 3D 28 04 :96
DD70 D6 05 30 09 2A 6E 2F E5 :C0
DD78 CD B1 28 E1 C9 3A 83 2F :3C
DD80 B7 CA 94 2D ED 5B 6E 2F :27
DD88 3A 61 2F 4F E6 0F FE 09 :15
DD90 38 0C CD 78 28 D2 94 2D :44
DD98 CB 79 20 24 18 D6 CD 78 :BB
DDA0 28 DA 94 2D CB 79 20 18 :3F
DDA8 18 CA 3A 83 2F B7 CA 94 :E3
DDB0 2D 11 00 00 CD 78 28 3E :E9
DDB8 98 30 02 3E 89 32 61 2F :53
DDC0 2A 57 2F 22 6E 2F D5 21 :65
DDC8 61 2F CD D0 0B E1 18 A7 :D8
DDD0 2A 4B 2F CD 60 2B 28 13 :37
DDD8 FE 28 20 0F CD 6A 2B 28 :DF
DDE0 0A FE 29 20 06 23 22 4B :E7
DDE8 2F 37 C9 B7 C9 DD E1 EB :58
DDF0 21 00 00 39 B7 ED 42 F9 :39
DDF8 EB ED B0 DD E9 DD E1 EB :F7
DE00 21 00 00 39 ED B0 F9 DD :CD
DE08 E9 3A 81 2F FE FF C8 2A :C2
DE10 4B 2F CD 60 2B 20 22 CD :E1
DE18 63 0B 30 0A CD EF 0A 38 :A6
DE20 F1 3E FF C3 ED 29 ED 53 :47
DE28 4D 2F E5 EB 3A 50 2F B7 :BC
DE30 C4 25 0B E1 CD 60 2B 28 :55
DE38 DE FE 2F 20 10 23 7E 2B :07
DE40 FE 2A 7E 20 08 23 CD 34 :F2
DE48 2B 38 D1 18 C5 22 5F 2F :C1
DE50 EB CD 6D 2B 38 05 CD 94 :EE
DE58 2B 18 3D FE 24 20 11 13 :E6
DE60 1A CD 74 2B 38 05 CD CC :5C
DE68 2B 18 2D 2A 57 2F 18 28 :60
DE70 FE 27 20 2B 26 00 13 1A :C3
DE78 B7 28 0A 6F 13 1A FE 27 :AA
DE80 20 0E 13 18 13 21 27 00 :B4
DE88 22 71 2F 21 20 00 18 08 :23
DE90 22 72 2F 3E 27 32 71 2F :FA
DE98 AF 22 5D 2F C3 E6 29 FE :2D
DEA0 5F 28 05 CD 86 2B 38 51 :93
DEA8 21 62 2F 06 0C 77 23 13 :71
DEB0 1A CD 6D 2B 30 09 CD 86 :0B
DEB8 2B 30 04 FE 5F 20 06 10 :F2
DEC0 EC 06 01 18 EA 36 00 ED :18
DEC8 53 4B 2F 11 FA 29 21 62 :84
DED0 2F 1A B7 FA 8C 29 28 1D :F4
DED8 4F 7E CD 57 2B B9 20 0C :01
DEE0 13 23 18 ED 7E B7 20 0A :9A
DEE8 1A C3 ED 29 13 1A B7 F2 :C9
DEF0 94 29 13 18 D9 3E 01 18 :18
DEF8 4C 21 19 2B 01 1B 00 ED :BA
DF00 B1 20 46 4F 13 ED 53 4B :04
DF08 2F FE 3A 20 07 1A FE 3D :E3
DF10 3E F0 18 27 FE 3C 20 0B :D2
DF18 1A D6 3C FE 03 30 25 C6 :48
DF20 F1 18 1A FE 3E 20 0B 1A :A4
DF28 D6 3D FE 02 30 16 C6 F4 :13
DF30 18 0B FE 40 20 0E 1A FE :A7
DF38 40 3E F6 20 07 13 ED 53 :EE
DF40 4B 2F 18 01 79 32 81 2F :EE
DF48 C9 6F 26 00 22 71 2F C3 :E3
DF50 1B 2D 41 4E 44 A8 41 54 :58
DF58 BE 42 52 45 41 4B 8B 42 :F0
DF60 59 9A 42 59 54 45 91 43 :FB
DF68 41 52 52 59 B4 43 4F 4E :D2
DF70 53 8C 44 41 54 41 8E 44 :CB
DF78 45 43 AD 44 45 43 4A B3 :FE
DF80 44 45 58 A5 44 45 42 55 :A6
DF88 47 84 45 4C 53 45 95 45 :CE
DF90 4C 53 45 49 46 BD 45 58 :CD
DF98 49 54 9E 46 4F 52 98 47 :01
DFA0 4F 9C 47 4F 54 4F 9D 48 :09
DFA8 49 BB 49 46 93 49 4E 43 :00
DFB0 AC 49 4E 43 4C 55 44 45 :B0
DFB8 81 49 4E 4C 49 4E 45 90 :D0
DFC0 49 4E 58 A4 49 58 BF 49 :3C
DFC8 59 C0 4C 44 58 A2 4C 4F :3E
DFD0 4F 50 9B 4C 4F 57 BC 4D :35
DFD8 45 4D 4F 52 59 B9 4D 49 :DB
DFE0 4E 55 53 AB 4E 4F 54 A9 :3B
DFE8 4F 52 A7 4F 56 45 52 46 :CA
DFF0 4C 4F 57 B8 50 41 52 49 :D6
DFF8 54 59 B7 50 4C 55 53 AA :52
E000 50 4F 52 54 BA 50 52 4F :F0
E008 47 80 52 45 43 55 52 53 :9B
E010 49 56 45 8F 52 45 54 55 :B3
E018 52 4E 9F 52 4C AE 52 4C :29
E020 43 AF 52 52 B0 52 52 43 :2D
E028 B1 53 45 54 A1 53 49 47 :21
E030 4E B6 53 52 41 B2 53 54 :43
E038 4F 50 A0 53 54 58 A3 54 :35
E040 48 45 4E 94 54 4F 99 54 :FF
E048 52 4F 46 46 86 54 52 4F :A8
E050 4E 85 55 4E 54 49 4C 97 :F6
E058 56 41 52 8D 57 48 49 4C :AA
E060 45 96 57 4F 52 44 92 58 :01
E068 4F 52 A6 5A 45 52 4F B5 :3C
E070 00 21 22 23 25 26 28 29 :02
E078 2A 2B 2C 2D 2E 2F 3A 3B :80
E080 3C 3D 3E 3F 40 5B 5D 5E :4C
E088 7B 7C 7D 7E 23 7E B7 28 :72
E090 14 FE 2A 20 F7 23 7E B7 :AB
E098 28 0B FE 2A 28 EF FE 2F :9F
E0A0 20 EA B7 23 C9 CD 63 0B :E8
E0A8 D8 ED 53 4D 2F 18 DE FE :88
E0B0 61 D8 FE 7B D0 D6 20 C9 :41
E0B8 7E B7 C8 FE 09 28 03 FE :2D
E0C0 20 C0 23 18 F3 FE 30 D8 :14
E0C8 FE 3A 3F C9 CD 6D 2B D0 :75
E0D0 FE 41 D8 FE 47 3F D0 FE :69
E0D8 61 D8 FE 67 3F C9 FE 41 :E5
```

```
E0E0 D8 FE 5B 3F D0 FE 61 D8 :77
E0E8 FE 7B 3F C9 21 00 00 1A :BC
E0F0 D6 30 FE 0A D0 44 4D 29 :98
E0F8 38 12 29 38 0F 09 38 0C :07
E100 29 38 09 4F 06 00 09 38 :00
E108 03 13 18 E3 2A 5F 2F 11 :DA
E110 71 2F 06 0F 7E CD 6D 2B :98
E118 38 49 05 04 28 03 12 13 :DA
E120 05 23 18 F0 21 00 00 1A :6B
E128 CD 74 2B D8 CD 57 2B D6 :69
E130 30 FE 0A 38 02 D6 07 4F :9E
E138 06 00 7C E6 F0 20 08 29 :A9
E140 29 29 29 09 13 18 E0 2A :B9
E148 5F 2F 11 71 2F 06 0E 7E :D1
E150 23 12 13 7E CD 74 2B 38 :6A
E158 0A 05 04 28 03 12 13 05 :68
E160 23 18 F0 E5 AF 12 CD 5D :FB
E168 2D D1 21 00 00 C9 11 1F :18
E170 2C CD 3A 2C 36 02 C9 76 :D6
E178 61 72 69 61 62 6C 65 00 :D0
E180 11 31 2C CD 3A 2C 36 01 :D8
E188 C9 63 6F 6E 73 74 61 6E :BF
E190 74 00 21 00 00 22 6E 2F :54
E198 21 4A 2C CD 56 0A 21 61 :46
E1A0 2F C9 23 4D 69 73 73 69 :20
E1A8 6E 67 20 5C 20 6E 61 6D :AD
E1B0 65 20 3A 20 40 62 2F 00 :B0
E1B8 21 66 2C C3 F5 2D 23 42 :FD
E1C0 61 64 20 6F 70 74 69 6F :10
E1C8 6E 20 73 77 69 74 63 68 :20
E1D0 00 21 7F 2C C3 F5 2D 23 :D4
E1D8 49 6C 6C 65 67 61 6C 20 :DA
E1E0 66 75 6E 63 74 69 6F 6E :66
E1E8 20 6E 61 6D 65 00 21 9F :81
E1F0 2C 11 62 2F C3 56 0A 23 :14
E1F8 40 80 2C 20 3A 20 5C 00 :C2
E200 21 AE 2C C3 F5 2D 23 49 :4C
E208 6C 6C 65 67 61 6C 20 6E :FF
E210 61 6D 65 00 21 C5 2C 11 :56
E218 62 2F C3 56 0A 23 40 AF :C6
E220 2C 20 3A 20 5C 00 11 62 :75
E228 2F 21 D7 2C C3 56 0A 23 :99
E230 49 6C 6C 65 67 61 6C 20 :DA
E238 6C 61 62 65 6C 20 3A 20 :7A
E240 5C 00 21 F0 2C C3 56 0A :BC
E248 23 42 61 64 20 73 74 72 :A3
E250 69 6E 67 20 64 61 74 61 :F8
E258 00 21 07 2D C3 56 0A 23 :9B
E260 54 6F 6F 20 6D 61 6E 79 :07
E268 20 61 72 67 75 6D 65 6E :0F
E270 74 73 00 21 24 2D 11 71 :DB
E278 2F C3 F5 2D 23 49 6C 6C :58
E280 65 67 61 6C 20 63 68 61 :E5
E288 72 61 63 74 65 72 20 3A :DB
E290 20 5C 00 11 40 2D 18 20 :32
E298 64 69 73 70 6C 61 63 65 :45
E2A0 6D 65 6E 74 00 11 52 2D :44
E2A8 18 0E 6F 76 65 72 20 72 :74
E2B0 61 6E 67 65 00 11 71 2F :4C
E2B8 21 6A 2D CD 56 0A 21 00 :06
E2C0 00 C9 23 49 6C 6C 65 67 :D9
E2C8 61 6C 20 63 6F 6E 73 74 :14
E2D0 61 6E 74 20 3A 20 5C 00 :19
E2D8 21 86 2D C3 F5 2D 23 42 :1E
E2E0 61 64 20 63 6F 6E 73 74 :0C
E2E8 61 6E 74 00 21 9A 2D C3 :EE
E2F0 F5 2D 23 42 61 64 20 61 :CD
E2F8 64 64 72 65 73 73 20 63 :08
E300 6F 6E 73 74 61 6E 74 00 :07
E308 21 B6 2D C3 F5 2D 23 53 :5F
E310 79 6E 74 61 78 20 65 72 :2B
E318 72 6F 72 00 21 CA 2D C3 :2E
E320 F5 2D 23 42 61 64 20 69 :D5
E328 6E 64 65 78 20 6F 70 65 :13
E330 72 61 74 69 6F 6E 00 21 :AE
E338 E5 2D C3 F5 2D 23 42 61 :BD
E340 64 20 65 78 70 72 65 73 :1B
E348 73 69 6F 6E 00 CD 56 0A :E6
E350 C3 44 0A C3 F2 2E C3 DA :91
E358 2E C3 22 2E C3 24 2E C3 :19
E360 34 2E C3 39 2E C3 3E 2E :BB
E368 C3 3F 2E C3 4F 2E C3 50 :83
E370 2E C3 60 2E C3 61 2E C3 :94
E378 72 2E 5E FE 5A 16 00 6A :D6
E380 67 3E 08 29 30 01 19 3D :5D
E388 20 F9 7D C9 CD 3E 2E 7B :13
E390 C9 CD 3F 2E 7B C9 56 5F :FC
E398 AF 2E 08 CB 23 17 BA 38 :DC
E3A0 02 92 1C 2D 20 F5 C9 56 :11
E3A8 14 15 C8 5F 7A FE 09 3E :0F
E3B0 00 D0 7B 87 15 20 FC C9 :CC
E3B8 56 14 15 C8 5F 7A FE 09 :27
E3C0 3E 00 D0 7B CB 3F 15 20 :C8
E3C8 FB C9 22 00 30 ED 53 FE :54
E3D0 2F E1 22 02 30 E1 22 04 :6B
E3D8 30 3A DA 2F B7 28 0D 21 :80
E3E0 DA 2F 5F 16 00 19 47 F1 :CF
E3E8 77 2B 10 FB 2A 04 30 E5 :F0
E3F0 2A FE 2F 7C B5 28 1E EB :B9
E3F8 21 00 00 ED 52 39 F9 C5 :57
E400 42 4B EB 2A 00 30 ED B0 :6F
E408 C1 2A 00 30 E5 2A FE 2F :57
E410 E5 21 D0 2E E5 79 B7 28 :41
E418 0B 2A 00 30 EB 21 DB 2F :7B
E420 06 00 ED B0 2A 02 30 E9 :E8
E428 C1 D1 21 00 00 39 ED B0 :89
E430 F9 C9 3A FB 2F B7 20 03 :00
E438 76 18 FD 3D 20 05 ED 7B :55
E440 FC 2F C9 3D 20 F2 2A FC :69
E448 2F E9 FC 2D FF 2D 02 2E :9D
E450 05 2E 08 2E 0B 2E 0E 2E :DE
E458 11 2E 14 2E 17 2E 1A 2E :0E
```

```
E460 1D 2E 20 2E 35 2E 3A 2E :64     E4B0 00 00 00 00 00 00 00 00 :00
E468 BA 2E 00 00 73 2E 77 2E :2E     E4B8 00 00 00 00 00 00 00 00 :00
E470 7B 2E 7F 2E 82 2E 88 2E :BC     E4C0 00 00 00 00 00 00 00 00 :00
E478 95 2E 99 2E AC 2E B2 2E :44     E4C8 00 00 00 00 00 00 00 00 :00
E480 B6 2E C2 2E C6 2E CD 2E :C3     E4D0 00 00 00 00 00 00 00 00 :00
E488 DB 2E E8 2E EF 2E 00 00 :3C     E4D8 00 00 00 00 00 00 00 00 :00
E490 00 00 00 00 00 00 00 00 :00     E4E0 00 00 00 00 00 00 00 00 :00
E498 00 00 00 00 00 00 00 00 :00     E4E8 00 00 00 00 00 00 00 00 :00
E4A0 00 00 00 00 00 00 00 00 :00     E4F0 00 00 00 00 00 00 00 00 :00
E4A8 00 00 00 00 00 00 00 00 :00     E4F8 00 00 00 00 00 00 00 00 :00
```

あとがき

本書で述べた内容は，コンパイラ作成の基本的なことなので，本格的なコンパイラをつくる場合には，いささか足りない部分があると思われます．さらに，詳しいことが知りたい方は参考文献に挙げた本を読むなり，実際に作動しているコンパイラのプログラム・リストを見たりして研究するのが望ましいでしょう．

しかし，何よりも実際に自分でコンパイラをつくってみることです．それが，仮に実用には向かないような貧弱なコンパイラであっても，完成するまでに得られる知識は万巻の書物を読むことより，はるかに多いように思われるからです．

■ 参考文献 ■

(1) Niklaus Wirth著，片山　卓也訳
『アルゴリズム＋データ構造＝プログラム』日本コンピュータ協会，1979

(2) 中西　正和，大野　義夫共著
『やさしいコンパイラの作り方』共立出版，1980

(3) 島内　剛一，広瀬　健，中田　育男，筧　捷彦，佐久間　紘一共著
『コンパイラのうちとそと』共立出版，1979

(4) 中田　育男著
『コンパイラの技法』竹内書店，1971

(5) Arthur B. Pyster著，松尾　正信訳
『コンパイラの設計と構築』近代科学社，1983

(6) F. R. A. Hopgood著，首藤　勝，関本　彰次共訳
『コンパイラの技法』サイエンス社，1973

(7) K. Jensen，N. Wirth共著，原田　賢一訳
『PASCAL』培風館，1981

(8) B. W. Kernighan，D. M. Ritchie共著，石田　晴久訳
『プログラミング言語C』共立出版，1981

(9) 上滝　致考編，戸田　英雄，榊原　清，矢田　光治共著
『入門FORTRAN77』オーム社，1983

(10) 國友　義久著
『効果的プログラミング開発技法』近代科学社，1979

(11) 『JISハンドブック　情報処理―1984』日本規格協会

(12) 矢田　光治著
『ソフトウェアの知識(第２版)』オーム社，1975

新言語作成の技法

昭和60年2月5日　初版発行　　定価3,000円

著　者　大貫広幸
発行者　塚本慶一郎
発行所　株式会社エム・アイ・エー
〒150 東京都渋谷区渋谷2-9-1 青山田中ビル
電　話　(03)486−4500(代表)

編集制作　アスキー出版局第二書籍編集部
電　話　(03)486−4512(直通)

印刷・製本　チトセ印刷

ISBN4-87170-029-1 C3055 ¥3000E

好評発売中

「入門MS-DOS」

村瀬康治著

定価1,500円(送料300円)

MS-DOS上のビジネスソフトなど、各種のソフトウェアを有効に使うために必要な基礎知識や、基本的な操作法を説明。豊富な実行例を基に解説していますので、MS-DOSをマスターするのに大きな効果がえられます。

内容：パソコンユーザーとMS-DOSとのかかわり　MS-DOSの起動　重要コマンドとその機能　MS-DOS上のソフトウェアetc.

「入門DISK BASIC」

戸内順一著

定価1,600円(送料300円)

DISK BASICを効果的に使用したいと考えているユーザーのための入門書。マイクロソフト系のDISK BASICの命令について、体系的に、しかもわかりやすい使用例を示しながら解説しています。

内容：DISK BASICとの出合い　DISK BASICを使う前に　プログラムを扱うファイル　ディスクに記録する形式etc.

「応用DISK BASIC」

戸内順一著

定価1,900円(送料300円)

DISK BASICを業務などで利用する場合のニーズに合わせて、ファイルやDISK BASICのコマンドを有効に利用するための数々の手法を、詳細に、しかも総合的に解説します。

内容：DISK BASICを超えて　論理レコードと物理レコード　サーチ法　順編成ファイル　直接編成ファイル 索引順次編成ファイル　ソート法etc.

株式会社アスキー

ASCII

お問い合せ、カタログ請求は 〒107東京都港区南青山5-11-5住友南青山ビルPHONE03(486)7111(代)